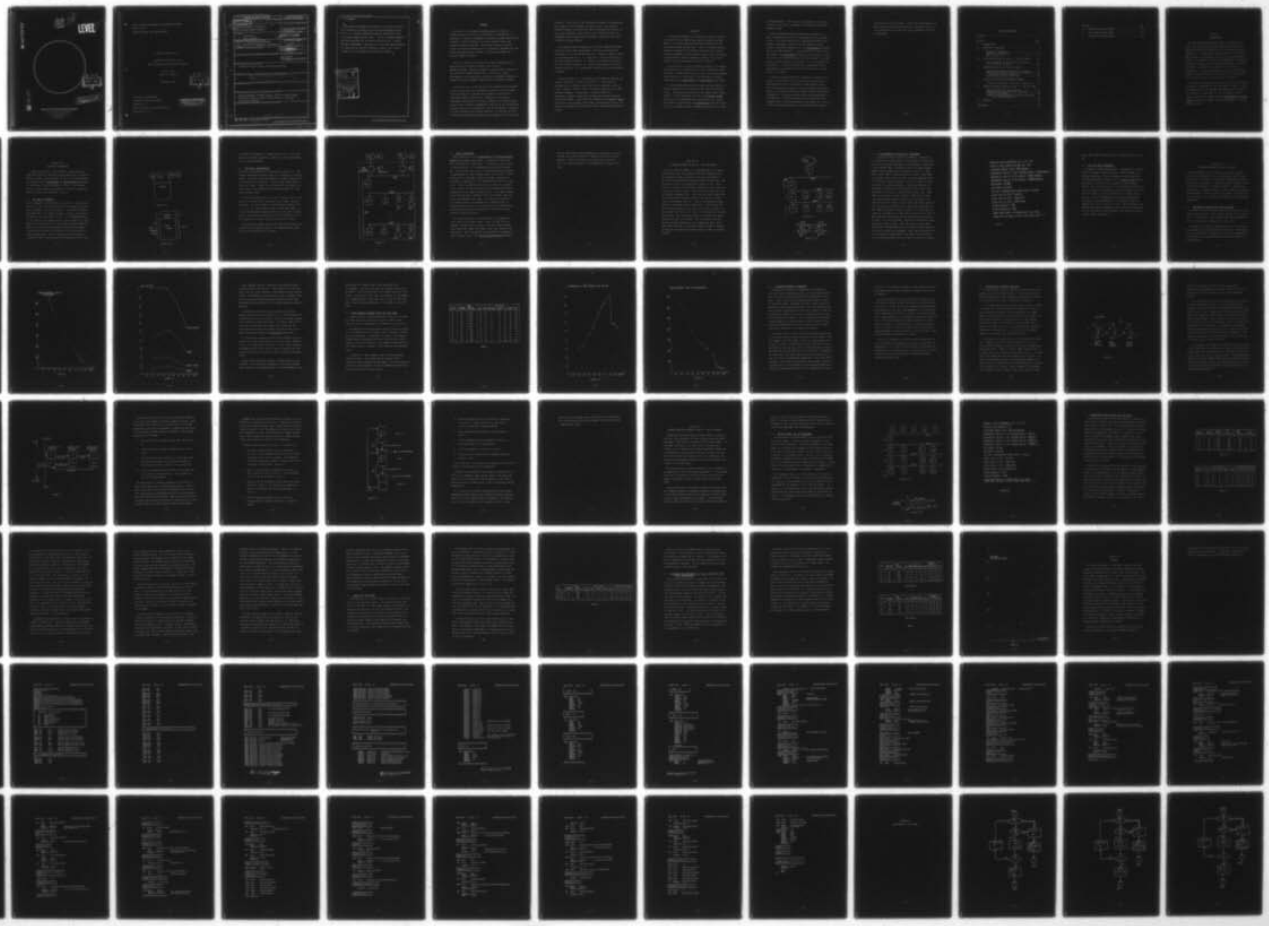
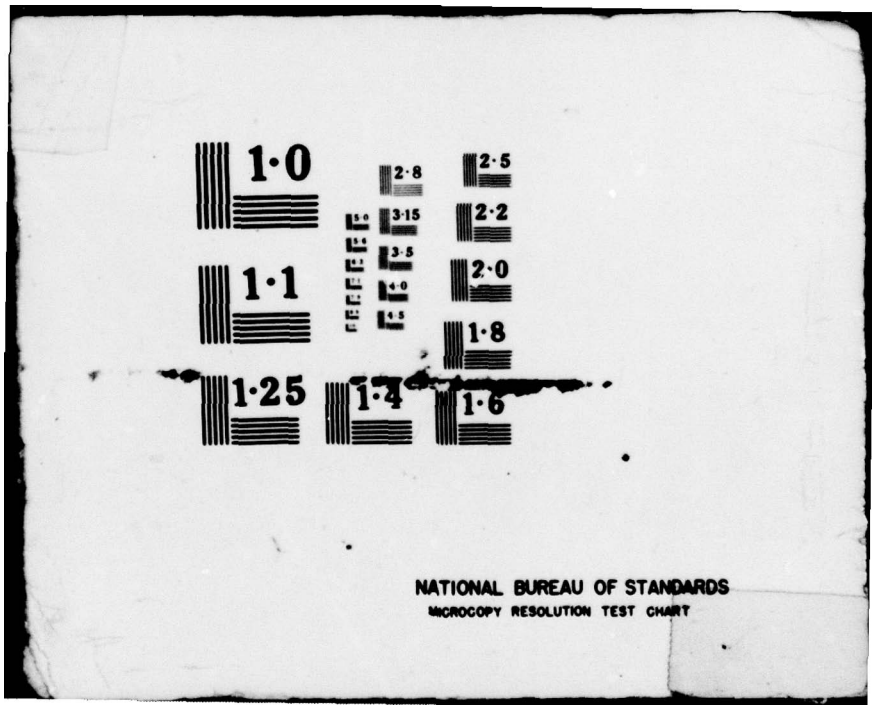
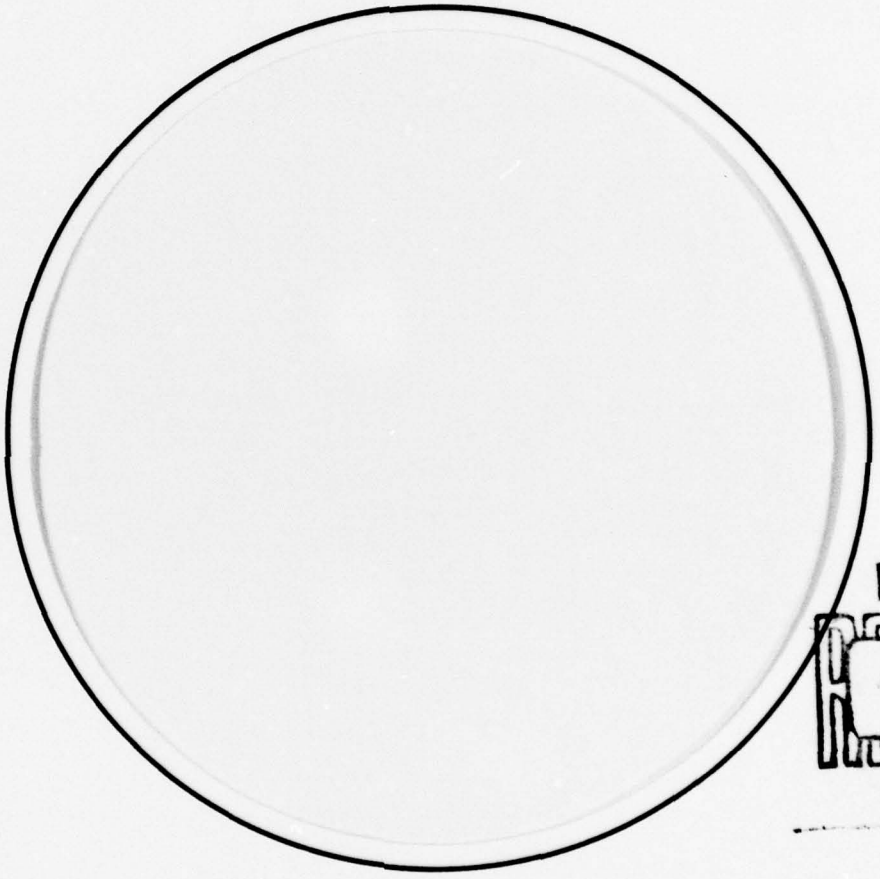
AD-A074 503 ALFRED P SLOAN SCHOOL OF MANAGEMENT CAMBRIDGE MA CEN--ETC F/G 9/2
SIMULATION STUDIES OF THE DSH-11 DATA STORAGE HIERARCHY SYSTEM.(U)
SEP 79 C LAM, S E MADNICK N00039-78-G-0160
UNCLASSIFIED CISR-M010-7909-04 NL

1 OF 2
AD-
A074503




1·0
1·1
1·25
2·8
3·15
3·5
4·0
1·4
2·5
2·2
2·0
1·8
1·6
NATIONAL BUREAU OF STANDARDS
MICROCOPY RESOLUTION TEST CHART



Center for Information Systems Research

Massachusetts Institute of Technology
Alfred P. Sloan School of Management
50 Memorial Drive
Cambridge, Massachusetts, 02139

617 253-1000

Basic Ordering Agreement No. N00039-78-G-0160

Task No. 003

Internal Report No. M010-7909-04

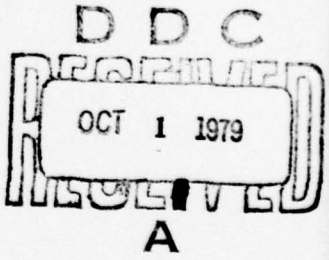
Technical Report No. 4

Simulation Studies of the DSH-11 Data Storage Hierarchy System

Chat-Yu Lam

Stuart E. Madnick

September 1979



Principal Investigator:

Professor Stuart E. Madnick

Prepared for:

Naval Electronics Systems Command

Washington, D.C.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

(9) 1. REPORT NUMBER: Technical Report, No. 4	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
(6) 4. TITLE (and Subtitle): Simulation Studies of the DSH-11 Data Storage Hierarchy System	5. TYPE OF REPORT & PERIOD COVERED
6. PERFORMING ORG. REPORT NUMBER: M010-7909-04	
(10) 7. AUTHOR(s): Chat-Yu Lam; Stuart E. Madnick	(15) 8. CONTRACT OR GRANT NUMBER(s): N00039-78-G-0160-001
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Center for Information Systems Research, M.I.T. Sloan School of Management, Cambridge, MA 02139	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: (12) 139 p.
11. CONTROLLING OFFICE NAME AND ADDRESS	(11) 12. REPORT DATE: September 1979
	13. NUMBER OF PAGES: 128
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

(14) CISR-M010-7909-04
CISR-TR-4

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

database management, database computer, multiple processor system, hierarchical system, C^3 system, storage hierarchy, simulation, performance evaluation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

409 590 JOB

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. ABSTRACT

This report discusses the results of a series of simulation studies of the DSH-11 Data Storage Hierarchy System. DSH-11 is a storage subsystem specially designed for the Intelligent Memory System (IMS). IMS is a backend computer specially designed for performing efficient and reliable database management. Descriptions of the IMS and the DSH-11 are presented in several companion reports. This report focuses on the performance evaluation of DSH-11.

Preface

The Center for Information Systems Research (CISR) is a research center of the M.I.T. Sloan School of Management. It consists of a group of management information systems specialists including : faculty members, full-time research staff, and student research assistants. The Center's general research thrust is to devise better means for designing, implementing, and maintaining application software, information systems, and decision support systems.

Within the context of the research effort sponsored by the Naval Electronics Systems Command under contract N00039-78-G-0160, CISR has proposed to conduct basic research on the Intelligent Memory System (IMS). The IMS is a high performance, high availability information management system for supporting future Command, Communication and Control Systems.

Current advances in LSI and Multi-Chip Integration technology offer the potential for development of modular multi-processor building blocks for information management, as well as for intelligent memory controllers. Advances in information management technologies have made it possible to hierarchically organize the information management functions so as to facilitate pipeline and parallel processing. The IMS attempts to integrate the above hardware and software advances. In the IMS, all the information management functions are decomposed into a functional

hierarchy. Each level of the functional hierarchy is implemented using modular multi-processor building blocks. An automatic storage hierarchy is used by the IMS for storage and retrieval of very large databases. Each level of the storage hierarchy is implemented using modular multi-processor controllers and their associated storage devices.

The proposed research described in Contract N00039-78-G-0160 focuses on the concept development, architectural design and evaluation of the IMS storage hierarchy. Specific research tasks to be accomplished are : (1) design of a general structure of the IMS storage hierarchy, (2) design of a revised structure of the IMS storage hierarchy, (3) develop algorithms for the IMS storage hierarchy, (4) performance evaluation of the IMS storage hierarchy.

Technical Report No. 1 introduces the concepts of IMS and its research directions. Technical Report No. 2 discusses the concepts of data storage hierarchies from a practical point of view. A design of DSH-1, the data storage hierarchy of the IMS database computer, is described. Technical Report No. 3 describes a simple structure of the IMS data storage hierarchy derived from DSH-1. Algorithms for supporting the read and write operations are developed. This report discusses the results of a series of simulation studies of the DSH-11 Data Storage Hierarchy System.

ABSTRACT

This report discusses the results of a series of simulation studies of the DSH-11 Data Storage Hierarchy System. DSH-11 is a storage subsystem specially designed for the Intelligent Memory System (IMS). IMS is a backend computer specially designed for performing efficient and reliable database management. Descriptions of the IMS and the DSH-11 are presented in several companion reports. This report focuses on the performance evaluation of DSH-11.

A key objective of these simulation studies is to assess the feasibility of supporting very large transaction rates (millions of reads and writes per second) with good response time (less than a millisecond) using the DSH-11 storage hierarchy and the read-through and store-behind algorithms.

An initial GPSS/360 simulation model was developed for a DSH-11 configuration with one processor and three storage levels. The results obtained from this model proved interesting. It was found that, at very high locality levels, when most of the references are satisfied by the highest performance storage level, the store-behind algorithm interacts with the DSH-11 buffer management algorithms to create

a system deadlock. This was not anticipated in the design of DSH-11, and led to a redesign of the DSH-11 buffer management scheme.

Another GPSS/360 simulation model was developed for a DSH-11 configuration with five processors and four storage levels. This model makes use of deadlock-free buffer management algorithms. Results from this model revealed further interesting properties of the store-behind algorithm and of the DSH-11 design. It was found that at high locality levels, the store-behind requests form a pipeline. Thus the rate of write operations that can be serviced is limited by the slowest stage in the pipeline, i.e., the slowest storage device. It was also found that a bottleneck may be developed at the lowest level when the block size of that level is too large.

A better balanced system was obtained by increasing the degree of parallelism in the lower storage levels and by decreasing the block sizes used by these storage levels. This system was then used as a basis to compare the performance of the DSH-11 architecture under different technology assumptions. It was found that using 1979 technologies, a throughput of .7 million operations per second with mean response time of 60 microseconds was obtainable for a mix of storage references consisting of 70 percent read requests

and 30 percent write requests. Using 1985 technologies, the same storage reference mix produced a throughput of 4 million operations per second with a mean response time of 10 microseconds.

TABLE OF CONTENTS

Appendix page

Section I

INTRODUCTION

The Intelligent Memory System (IMS) is a high performance, high availability database computer structured as a functional hierarchy interfacing with a storage hierarchy (Figure 1.1). Concepts and research issues of the IMS Data Base Computer are discussed in detail in (Lam and Madnick, 1979a). The approach of IMS is to decompose the information management functions into a hierarchy of modules. Parallel and pipeline operations are exploited to obtain high throughput. A storage hierarchy is specifically designed to support multiple processor and acts as a very large permenant virtual storage for the functional hierarchy.

A design of the general structure of the IMS Data Storage Hierarchy is described in (Lam and Madnick, 1979b). A simplified design of the IMS Data Storage Hierarchy, referred to as DSH-11, is described in (Lam and Madnick, 1979c). Detail protocols for supporting the read-through and store-behind operations in DSH-11 are also presented in (Lam and Madnick, 1979c).

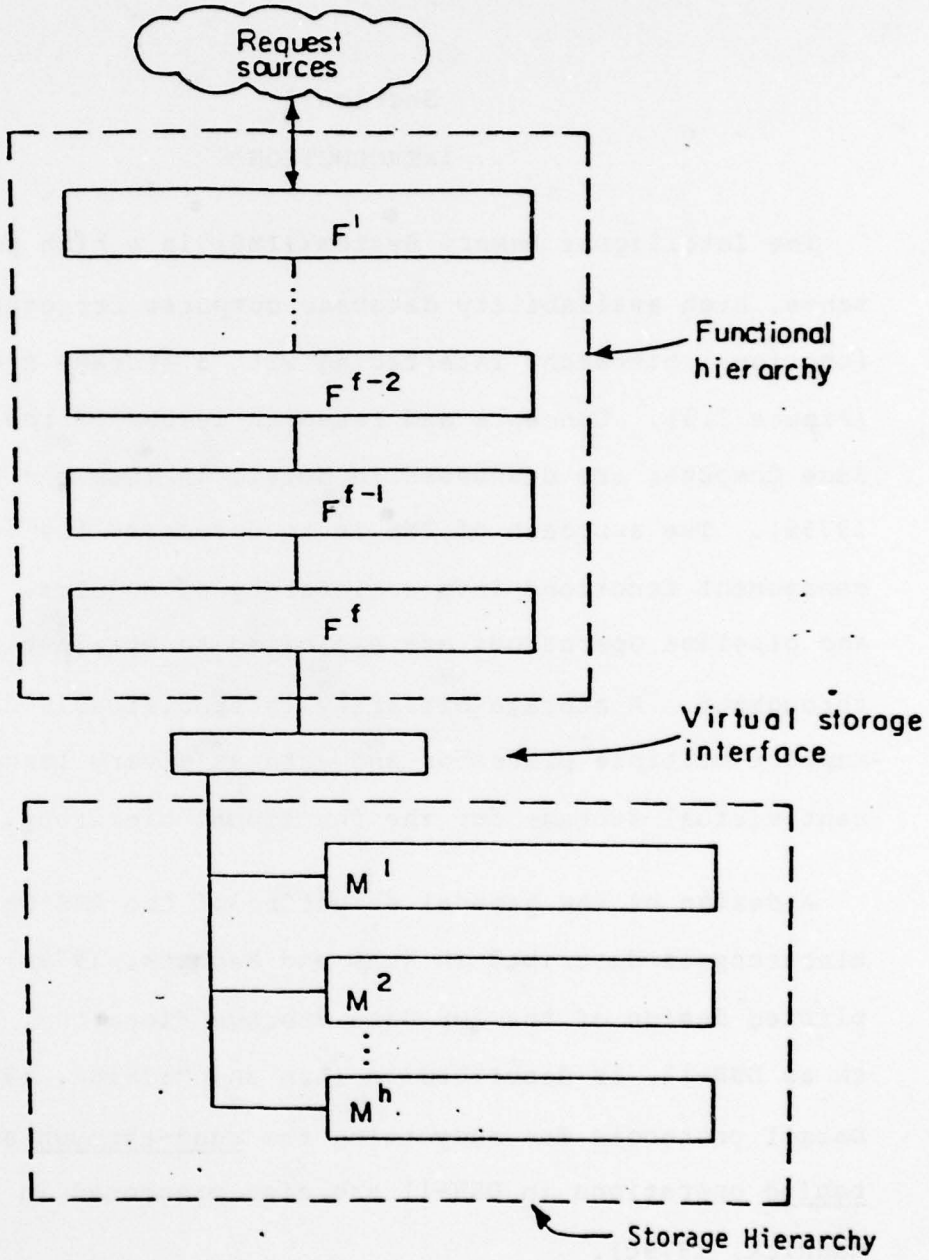

Request sources
F^1
F^{f-2}
F^{f-1}
F^f
Functional hierarchy
Virtual storage interface
M^1
M^2
M^h
Storage Hierarchy

Figure 1.1

This report is concerned with the performance of DSH-11. Simulation studies have been carried out to answer some basic questions about the performance of DSH-11. It was also hoped that these studies would provide some insights to the read-through and store-behind algorithms.

In the following sections, the DSH-11 structure and its algorithms will be briefly reviewed. Then a simulation model of DSH-11 with one processor and three storage levels, referred to as the P1L3 model, is described. Simulation results obtained from this model are analyzed. Then another simulation model of DSH-11 with five processors and four storage levels, referred to as the P5L4 model, is developed. This model is then used to assess the capabilities of DSH-11 under different technology assumptions.

Section II

THE DSH-11 STRUCTURE

A general structure of the IMS data storage hierarchy from which DSH-11 is derived has been described in (Lam and Madnick, 1979b). The structure of DSH-11 and the protocols for supporting the read-through and store-behind operations are described in (Lam and Madnick, 1979c). This section briefly reviews materials presented in detail in (Lam and Madnick, 1979b; Lam and Madnick, 1979c).

2.1 THE DSH-11 INTERFACE

DSH-11 supports a large number of processors. Each processor has its own instruction memory and uses DSH-11 as a very large permenant virtual data memory. A processor directly addresses DSH-11 via address mapping. This is illustrated in Figure 2.1(a) and Figure 2.1(b). If the addressed data is not found in the highest storage level of DSH-11, a delay similar to a page fault in virtual memory systems (Denning, 1970) will occur and the processor may switch to another process while waiting for the data to be retrieved from a lower storage level. Thus, a processor may have several requests to DSH-11 pending and many requests can be at vari-

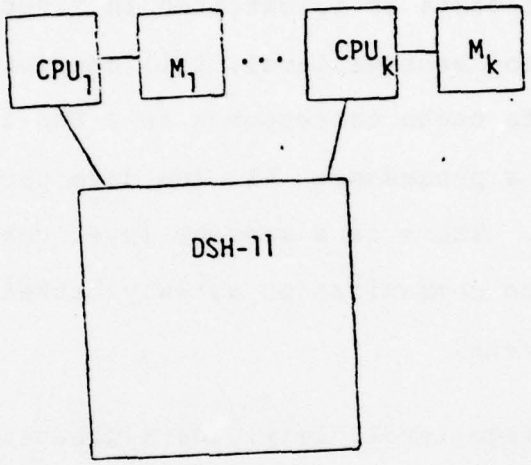


Figure 2.1(a)

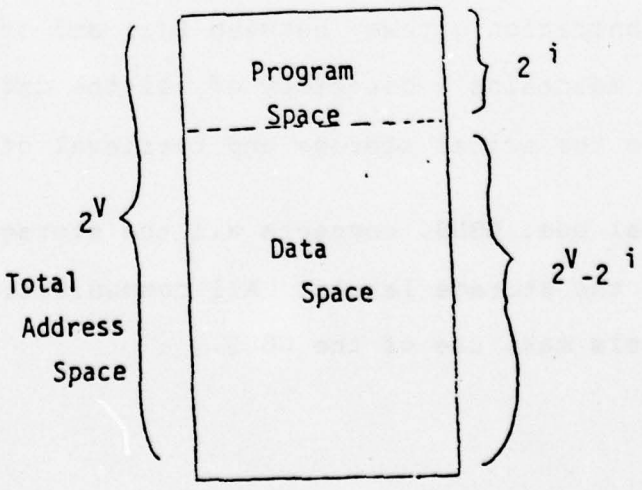


Figure 2.1(b)

ous stages of completion in DSH-11 at one time. This high degree of parallel operation in DSH-11 is a key determinant of its high performance.

2.2 THE DSH-11 ARCHITECTURE

The DSH-11 architecture is illustrated in Figure 2.2. The highest performance storage level, L(1) consists of the data caches. Each data cache corresponds to a DSH-11 memory port that connects to a processor. All the data caches share a local bus, LBUS1. There is a storage level controller, K1, that serves as the communication gateway between L(1) and lower storage levels.

A typical storage level, L(i), for i greater than 1, consists of a storage level controller, Ki, a memory request processor, Ri, and a number of storage device modules, Di1, ... , Dim. All these modules share a local bus, LBUSi. Ki is the communication gateway between L(i) and other storage levels. Ri maintains a directory of all the data in L(i). Dij performs the actual storage and retrieval of data.

The global bus, GBUS, connects all the storage level controllers of the storage levels. All communications among storage levels make use of the GBUS.

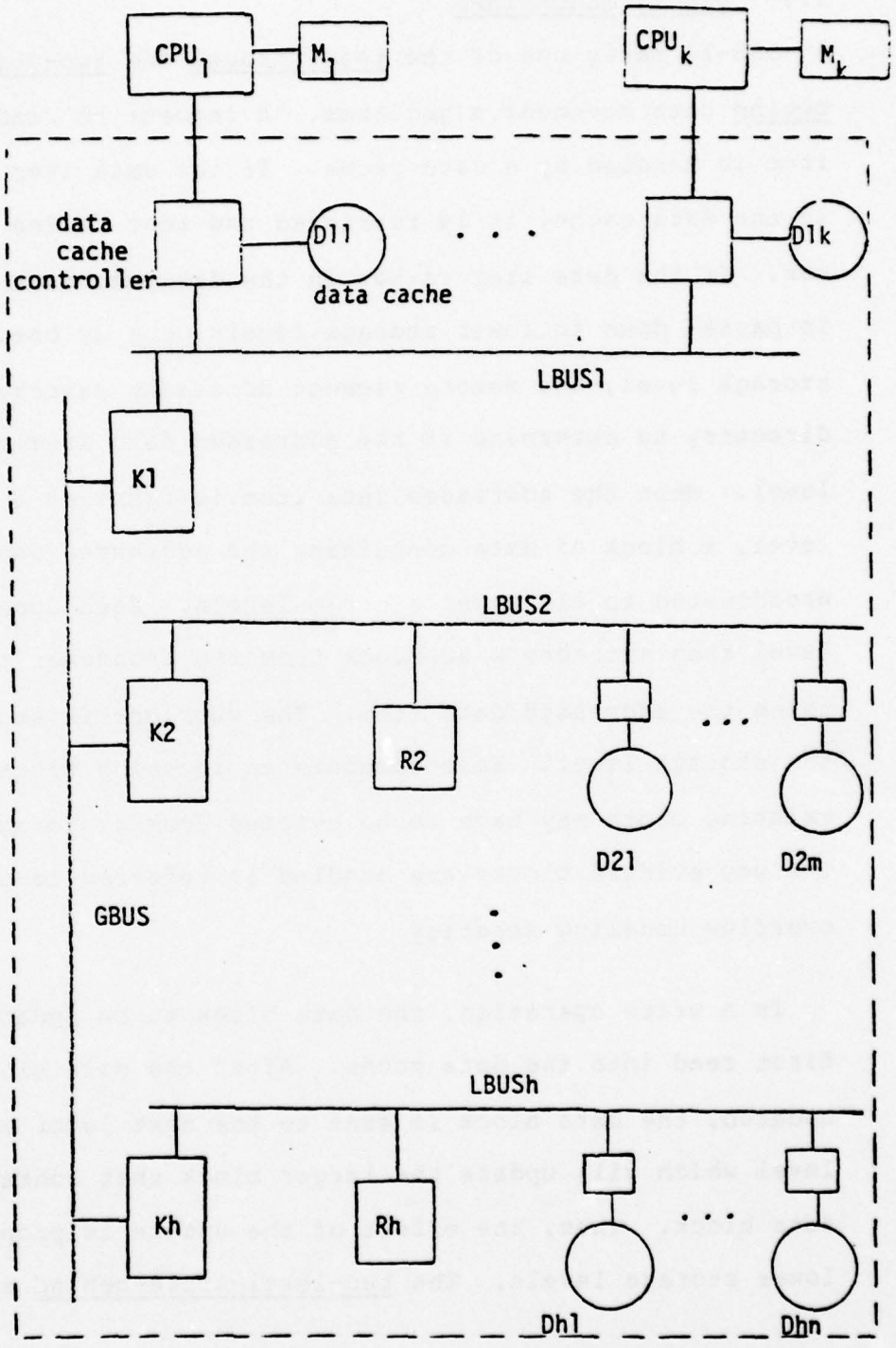

CPU1
M1
CPUk
Mk
data
cache
controller
D11
D1k
data cache
LBUS1
K1
LBUS2
K2
R2
D21
D2m
GBUS
LBUSh
Kh
Rh
Dh1
Dhn

Figure 2.2

2.3 DSH-11 ALGORITHMS

DSH-11 makes use of the read-through and two-level-store-behind data movement algorithms. A request to read a data item is handled by a data cache. If the data item is found in the data cache, it is retrieved and sent to the processor. If the data item is not in the data cache, the request is passed down to lower storage levels, one by one. At each storage level, the memory request processor searches its directory to determine if the addressed data item is in that level. When the addressed data item is found at a storage level, a block of data containing the addressed data item is broadcasted to all upper storage levels. Each upper storage level then extracts a subblock from the broadcast that contains the addressed data item. The subblock is stored in the storage level. To accomodate an incoming block, an existing block may have to be evicted from a storage level. The way evicted blocks are handled is referred to as the overflow handling strategy.

In a write operation, the data block to be updated is first read into the data cache. After the data block is updated, the data block is sent to the next lower storage level which will update the larger block that contains the data block. Thus, the effect of the update is propagated to lower storage levels. The two-level-store-behind strategy

ensures that proper acknowledgements are obtained at a given storage level that indicates an updated block has been propagated at least two storage levels down the hierarchy. Thus, at least two copies of the updated data exists at all times.

Section III

A SIMULATION MODEL OF DSH-11 : THE P1L3 MODEL

The P1L3 model of DSH-11 is a GPSS/360 model of a DSH-11 configuration with one processor and three storage levels. It represents a basic structure from which extensions to include more processors and storage levels can be made. The structure of P1L3 is illustrated in Figure 3.1(a). Each module in Figure 3.1(a) actually consists of four queues and a facility (Figure 3.1(b)). The facility is referred to as the request processor (RP). There are two input queues, one for transactions with data (the XQ), and one for transactions with messages (the IQ). The two corresponding output queues are named YQ and OQ respectively. The XQs and YQs have limited capacity, since they are the data buffers. We will assume that there is no limit on the lengths of the IQs and the OQs. The following example illustrates the naming conventions used in the model. The K2 storage level controller actually consists of five components, KRP2, KIQ2, KOQ2, KXQ2 and KYQ2. The current length of KXQ2 is denoted as KXL2 and the maximum allowable length of KXQ2 is denoted as KXM2.

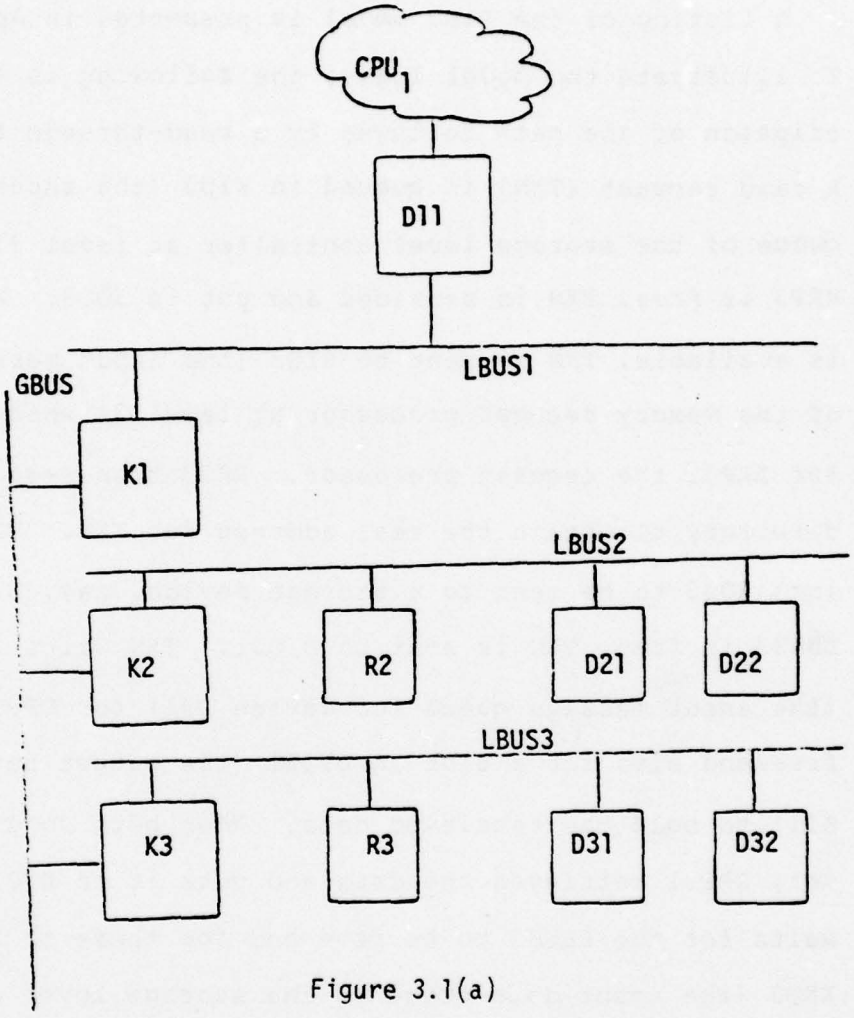


Figure 3.1(a)

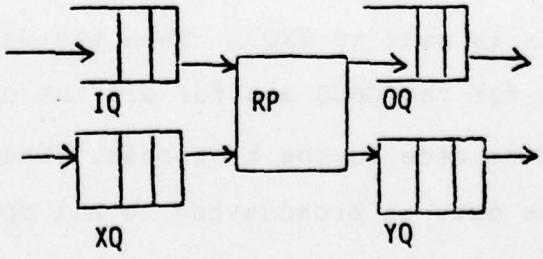


Figure 3.1(b)

3.1 AN ILLUSTRATION OF THE DSH-11 ALGORITHMS

A listing of the PlL3 model is presented in Appendix A. To illustrate the model logic, the following is a brief description of the path followed by a read-through transaction. A read request (TXN) is queued in KIQ3 (the input message queue of the storage level controller at level 3). When KRP3 is free, TXN is serviced and put in KOQ3. When LBUS3 is available, TXN is sent to RIQ3 (the input message queue of the memory request processor at level 3) where it waits for RRP3, the request processor. RRP3 then searches its directory to obtain the real address for TXN. TXN is put into ROQ3 to be sent to a storage device, say, D31. When LBUS3 is free, TXN is sent to DIQ31. TXN waits in DIQ31 (the input message queue for device D31) for DRP31 to be free and also for a slot in DYQ31 (the output data queue for D31) to hold the retrieved data. When both conditions are met, DRP31 retrieves the data and puts it in DYQ31 where it waits for the LBUS3 to be free and for there to be a slot in KXQ3 (the input data queue of the storage level controller at level 3) to hold the data. When both conditions are met, the data is sent to KXQ3. Then the data is put in KYQ3 waiting for the GBUS and for all the upper storage levels to be free to receive the broadcast. When these conditions are met, the data is broadcasted to all upper storage levels. At the highest storage level, the data is sent to the appro-

DEGREE OF MULTIPROGRAMING OF A CPU = 20

SIZES OF DATA QUEUES (XQ AND YQ) = 10

DIRECTORY SEARCH TIME = 200 NANOSEC.

READ/WRITE TIME OF A L(1) STORAGE DEVICE = 100 NANOSEC.

READ/WRITE TIME OF A L(2) DEVICE = 1000 NANOSEC.

READ/WRITE TIME OF A L(3) DEVICE = 10000 NANOSEC.

BUS SPEED = 10 MHZ

BUS WIDTH = 8 BYTES

SIZE OF A TRANSACTION WITHOUT DATA = 8 BYTES

BLOCK SIZE AT L(1) = 8 BYTES

BLOCK SIZE AT L(2) = 128 BYTES

BLOCK SIZE AT L(3) = 1024 BYTES

% READ REQUESTS = 70%

% WRITE REQUESTS = 30%

CONDITIONAL PROB. OF FINDING DATA IN A LEVEL

GIVEN THAT THE DATA IS NOT IN ANY UPPER LEVEL = P

Figure 3.2

priate data cache controller which forwards the data to the CPU.

3.2 THE P1L3 MODEL PARAMETERS

The model is highly parametized. Parameters for the P1L3 model are chosen to reflect current (1979) processor and storage technology. Two key parameters that characterize the references made to DSH-11 are the locality level and the proportion of read and write requests in the reference stream. The locality level (P) is the condition probability that a reference is satisfied at a given storage level given that the reference is not satisfied in all upper storage levels. Figure 3.2 summarizes all the model parameters. The degree of multiprogramming is the maximum number of requests that can be active at a CPU. The block sizes, bus speeds, bus width and speeds of the devices are parametized. For the P1L3 model, these parameters are chosen to reflect current (1979) technology.

Section IV

SIMULATION RESULTS OF THE P1L3 MODEL

Three different locality levels are used for the P1L3 model. The simulated time is one milisecond (one million time units in the model). Some unusual phenomena are uncovered during the analysis of these preliminary results. This leads to more extensive simulation studies to obtain more data points. A plausible theory is then proposed to explain these phenomena. Detail traces of the model is used to verify the theory. The findings are discussed in the following subsections.

4.1 PRELIMINARY STUDIES USING THE P1L3 MODEL

A series of three simulation studies are carried out with three locality levels : high (P=.85), medium (P=.5), and low (P=.2). Throughputs, mean response times and utilizations of the facilities are summarized in Figure 4.1.

Throughput in millions transactions per second are plotted against the locality levels in Figure 4.2. From Figure 4.2, it seems that a throughput of .6 million transactions per second is the maximum that one could obtain with this configuration.

Locality Level (P)	Throughput (per msec)	Mean Response Time (NSec)	Utilizations						
			GBUS	LBUS1	DATA CACHE	LBUS2	D21	LBUS3	D31
.20	285	64032	.41	.07	.10	.63	.11	.52	.52
.50	548	31324	.50	.09	.19	.65	.17	.62	.99
.85	598	6021	.23	.05	.19	.26	.09	.35	1.00

FIGURE 4.1

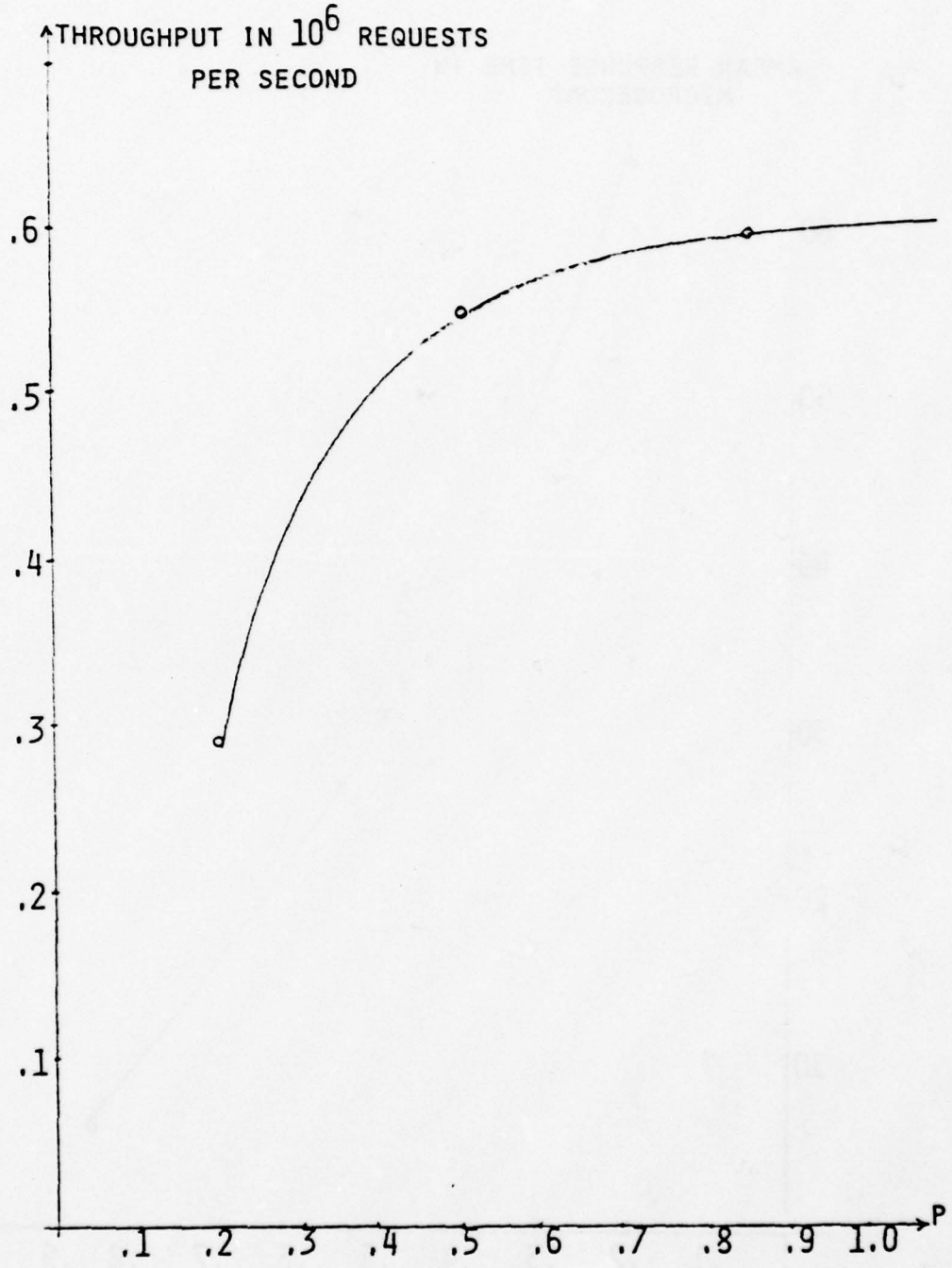

THROUGHPUT IN 10^6 REQUESTS PER SECOND
.6
.5
.4
.3
.2
.1
.1
.2
.3
.4
.5
.6
.7
.8
.9
1.0
P

Figure 4.2

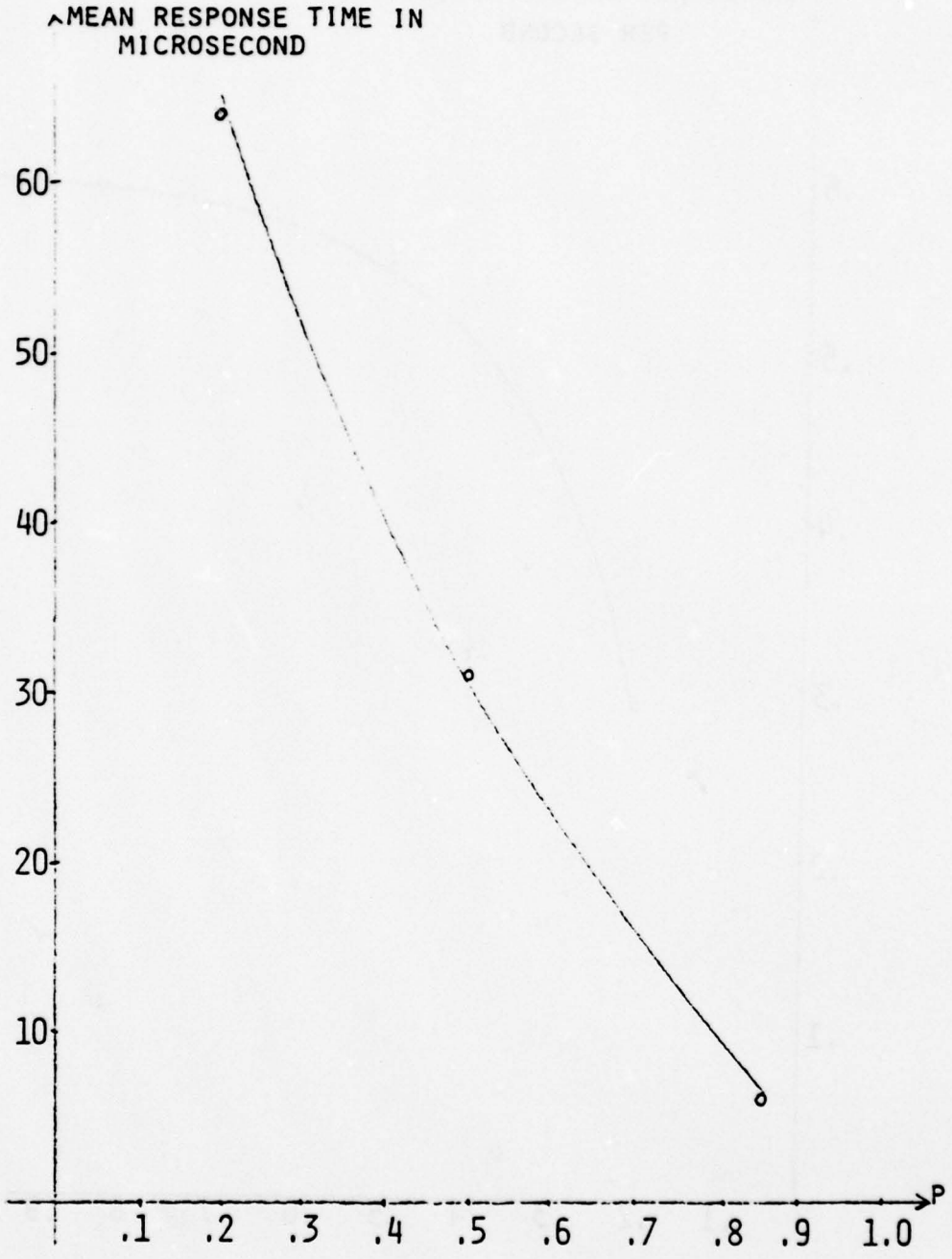

^MEAN RESPONSE TIME IN MICROSECOND
60
50
40
30
20
10
.1
.2
.3
.4
.5
.6
.7
.8
.9
1.0
P

Figure 4.3

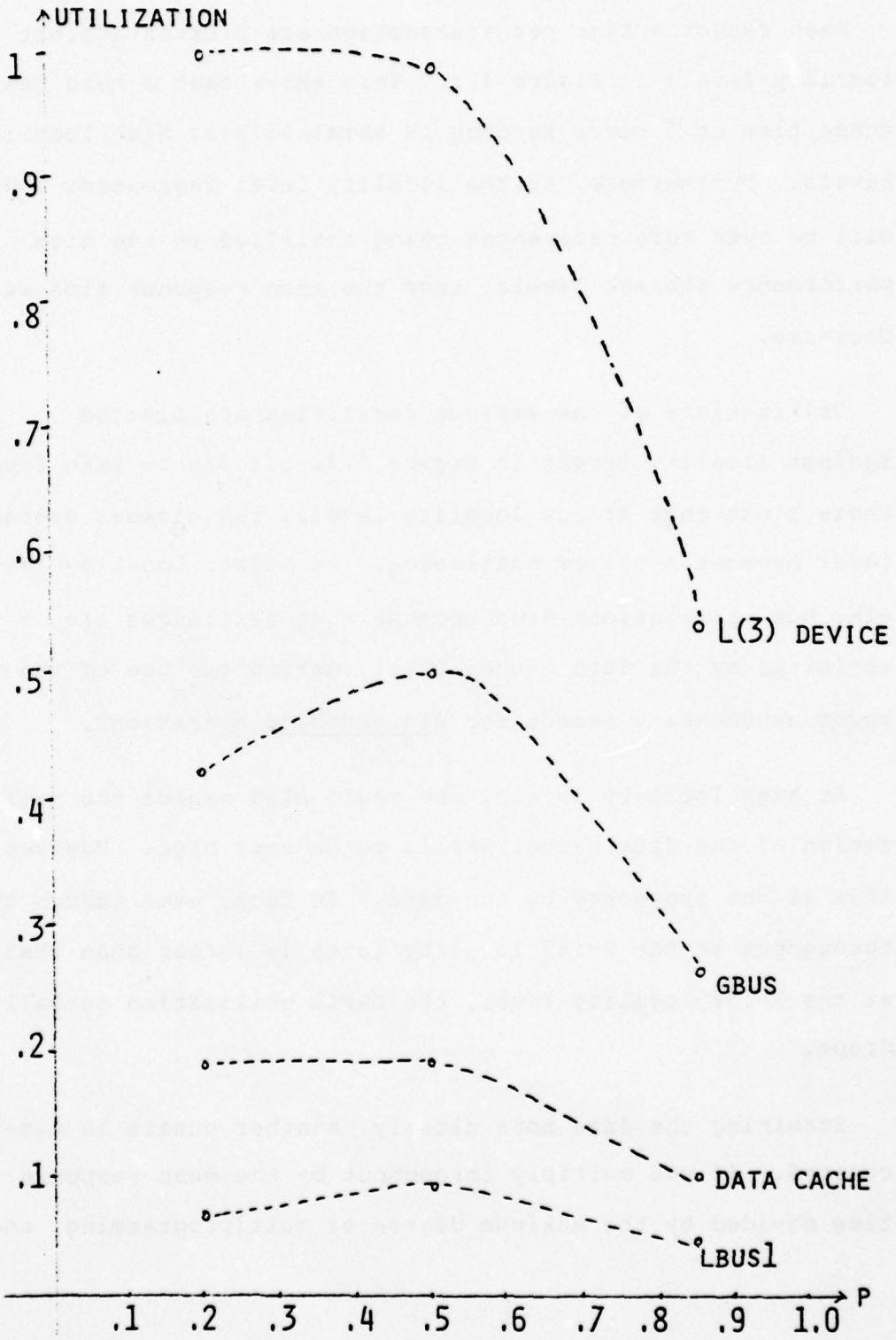

UTILIZATION
1
.9
.8
.7
.6
.5
.4
.3
.2
.1
L(3) DEVICE
GBUS
DATA CACHE
LBUS1
.1
.2
.3
.4
.5
.6
.7
.8
.9
1.0
P

Figure 4.4

Mean response time per transaction are plotted against locality levels in Figure 4.3. This shows that a mean response time of 5 micro seconds is obtainable at high locality levels. Furthermore, as the locality level increases, there will be even more references being satisfied in the high performance storage levels, thus the mean response time will decrease.

Utilizations of the various facilities are plotted against locality levels in Figure 4.4. It can be seen from these plots that at low locality levels, the slowest storage level becomes a system bottleneck. At higher locality levels, bus utilizations drop because most references are satisfied by the data cache, DRP11, making the use of the buses unnecessary except for store-behind operations.

At high locality levels, one would also expect the utilization of the data cache, DRP11, to be very high. However, this is not supported by the data. In fact, even though the throughput at the P=.85 locality level is larger than that at the P=.50 locality level, the DRP11 utilization actually drops.

Examining the data more closely, another puzzle is discovered. If you multiply throughput by the mean response time divided by the maximum degree of multiprogramming, one

should obtain a number close to the simulated time (1,000,000). For the P=.20 case, this number comes out to be 915657. For the P=.50 case, this number comes out to be 858277. But for the P=.85 case, this number is only 180027. It is suspected that either the data is wrong or there are some unusual blocking phenomena in the system in the P=.85 case.

4.2 MORE EXTENSIVE STUDIES USING THE P1L3 MODEL

A second series of simulations was carried out to obtain more data points by varying the locality levels. The results of these simulations are presented in Figure 4.5.

Throughputs are plotted against locality levels in Figure 4.6. In general, as the locality level increases, throughput also increases. A throughput of close to one million transactions is obtainable at about P=.80 locality level. However, after the P=.80 point, throughput drops sharply as the locality level increases. This requires some explaination.

In Figure 4.7, mean response time is plotted against locality levels. This shows that as locality level increases, mean response time decreases. This plot does not seem to provide insight as to why throughput decrease sharply after the P=.80 locality level.

Locality Level (P)	Throughput (per msec)	Mean Response Time (NSec)	Utilizations						
			GBUS	LBUS1	DATA CACHE	LBUS2	D21	LBUS3	D31
.20	286	64032	.42	.07	.10	.63	.11	.52	1.00
.30	320	56908	.42	.07	.12	.60	.12	.52	1.00
.40	456	39142	.45	.08	.16	.63	.15	.59	1.00
.50	548	31324	.50	.10	.20	.65	.17	.62	1.00
.60	698	27114	.51	.10	.23	.63	.19	.65	1.00
.65	758	22505	.51	.10	.26	.62	.18	.68	1.00
.70	811	23317	.53	.10	.27	.65	.20	.69	1.00
.80	947	16298	.50	.94	.31	.57	.19	.71	.99
.85	598	6021	.23	.52	.19	.26	.09	.35	.52
.90	581	3957	.16	.04	.17	.19	.06	.26	.42
.95	532	3986	.14	.03	.15	.16	.05	.21	.25

FIGURE 4.5

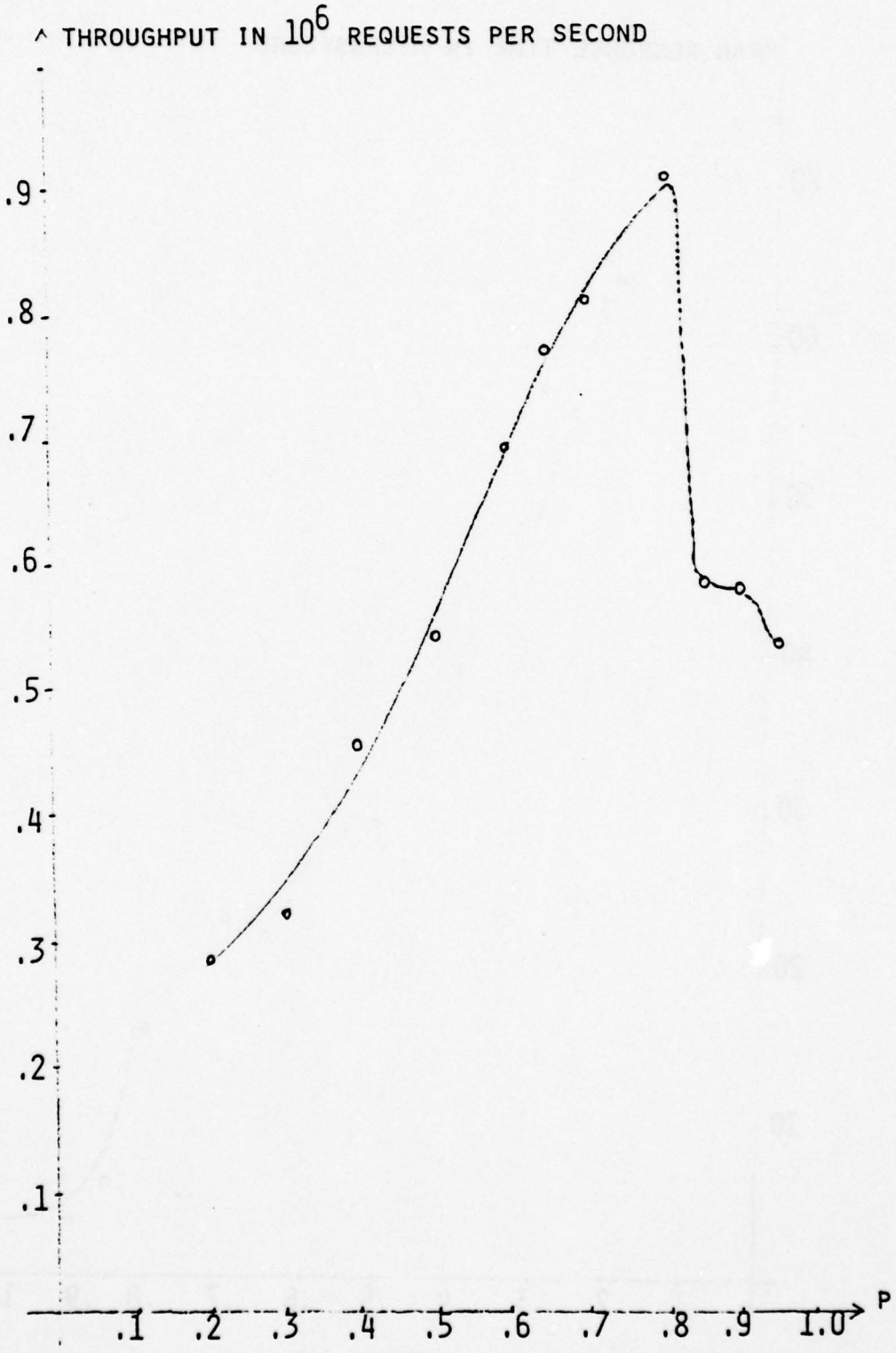

^ THROUGHPUT IN 10^6 REQUESTS PER SECOND
.9
.8
.7
.6
.5
.4
.3
.2
.1
.1
.2
.3
.4
.5
.6
.7
.8
.9
1.0
P

Figure 4.6

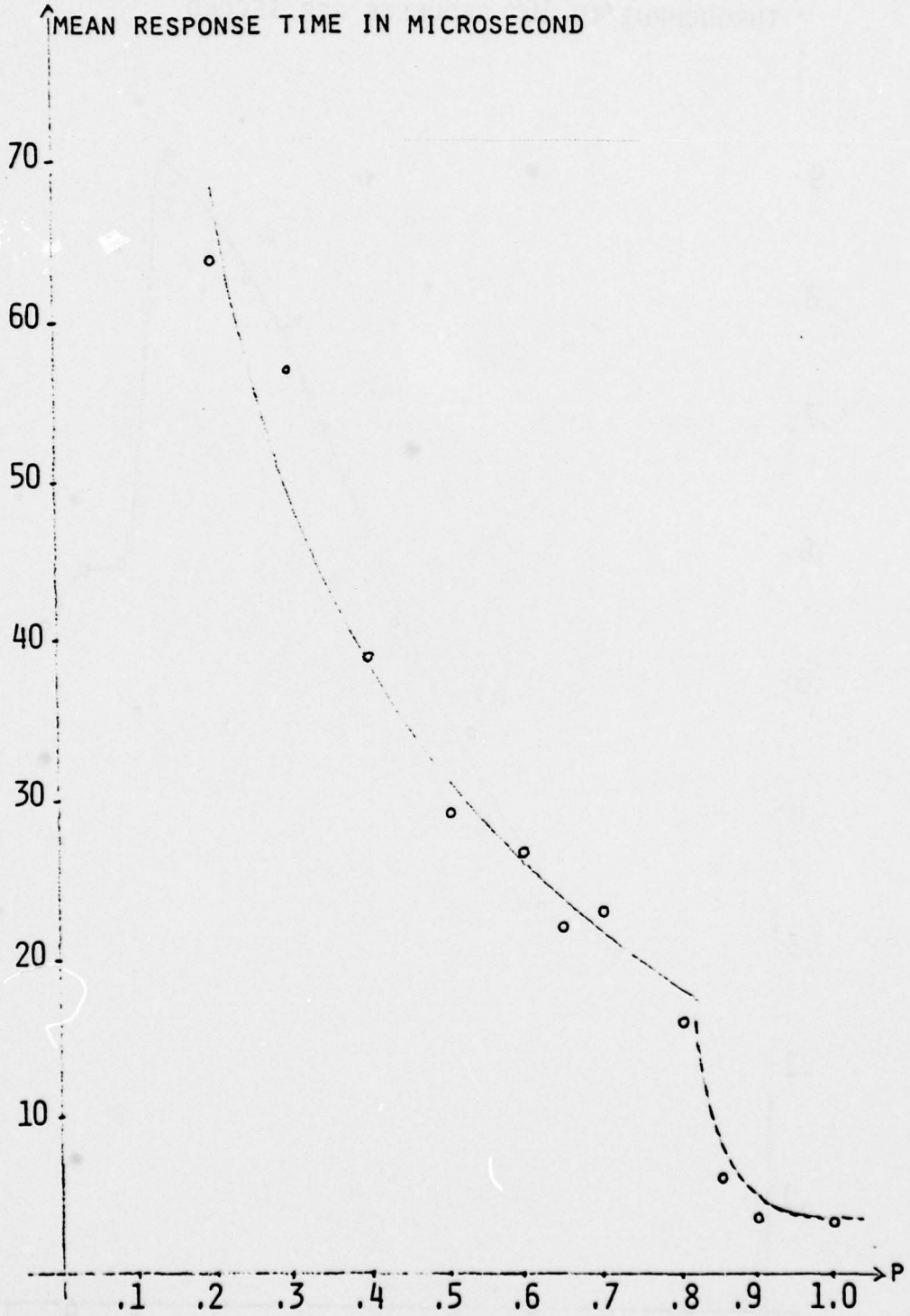

MEAN RESPONSE TIME IN MICROSECOND
70
60
50
40
30
20
10
.1
.2
.3
.4
.5
.6
.7
.8
.9
1.0
P

Figure 4.7

4.3 A PLAUSIBLE THEORY OF OPERATION

One theory to explain the sharp drop in throughput at very high locality levels is that at such high locality levels, the rate of write operations being generated is very high. Since a write will not proceed until DRP11 is free (to write the data), and DRP11's YQ has a buffer slot (for holding the store-behind operation), the write operation may hold up other transactions in their use of DRP11. Since the utilization of DRP11 is very low, the blocking must be due to the YQ being full often. Many store-behind transactions closed togther will tend to make the YQ full often. These blocking transactions will tend to hold up other transactions hence resulting in low system throughput.

If the YQ is full often, it must be because transactions in it cannot move on to the next facility fast enough. This will happen if the bus LBUS1 is busy or the XQ buffer of K1 is full, or both. From the data, we see that all the bus utilizations are very low, hence the blocking must be due to the fact that the XQ buffer of K1 is full often. Proceeding in this manner, one could argue that at high locality levels, the rate of store-behind operations is very high, which results in store-behind transactions being backed up from a storage device. This backing up of store-behind operations causes long queueing delays for other transactions as well,

resulting in low system throughput. This blocking situation also slows down the usage of DRP11 as evident from its low utilization.

We can now explain why the utilization of DRP11 at the P=.85 locality level is lower than that at the P=.50 locality level. At P=.85, due to the store-behind transactions being backed up, very few acknowledgements to the store-behind transactions ever return to DRP11. In the P=.50 case, most acknowledgements to store-behind transactions return to DRP11. Thus, even though the number of reads and writes handled by DRP11 in the P=.50 case is lower than that handled by the DRP11 in the P=.85 case, there are many more acknowledgements serviced by DRP11 in the P=.50 case, hence the corresponding utilization is higher.

There is no backing up of store-behind transactions in the low locality levels because the rate at which they are generated is low. Since the store-behind transactions are separated from one another there is enough time for a device to service a previous store-behind transaction before another one comes along.

4.4 VERIFICATION OF THEORY WITH DATA

The above theory seems to explain the phenomena well and agrees well with the observed data. To verify the theory, detailed model traces were examined to determine the status of the system at the time of simulation termination.

It is found that for low locality levels, there is indeed no backing up of the store-behind transactions. There is only a backlog of requests to be processed by the lowest storage level devices due to their large service times. For high locality levels, starting from P=.85, store-behind transactions begin to be backed up, from storage level 2. However, the back up is due to a system deadlock situation developed at storage level 2, and not due to the slower speeds of the devices, as hypothesized above.

The deadlock at storage level 2 is illustrated in Figure 4.8. Assume that all the XQs and YQs are full: (1) A store-behind transaction in DYQ21 is waiting for LBUS2 and a KXQ2 buffer slot. LBUS2 is free but KXQ2 buffer is full. (2) KXQ2 will not be cleared because KYQ2 is full. (3) KYQ2 cannot be cleared because both buffers of R2 are full. (4 and 5) These buffers cannot be cleared because DXQ21 and DYQ21 are full. DYQ21 cannot be cleared because it is waiting for KXQ2 to be cleared. Thus a deadlock is developed. This deadlock causes the XQs and YQs in the upper storage

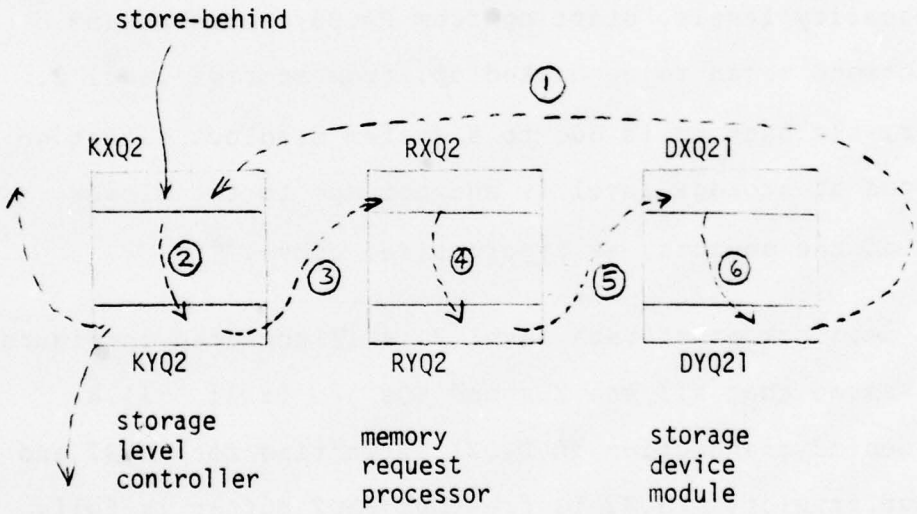

store-behind
KXQ2
RXQ2
DXQ21
KYQ2
RYQ2
DYQ21
storage
level
controller
memory
request
processor
storage
device
module

Figure 4.8

levels to be gradually filled as more store-behind transactions are generated. When YQ at DRP11 is full, the system will be held up when the next write transaction arrives.

It is interesting to note that this deadlock only occurs at very high locality levels. This is because at high locality levels, the rate of store-behind transactions generated is very high. Comparing the P=.95 case and the P=.50 case, even though the same number of store-behind transactions are generated to lower storage levels in both cases, the rate at which they are generated in the P=.95 case is 30 times that of the P=.50 case. Store-behind transactions sparsely separated from one another give chance for the device to service them, therefore avoiding a deadlock. This deadlock situation is not too different from a traffic jam at a Boston rotary during rush hour.

In retrospect, the causes of the deadlock are due to the rate of store-behind transactions and due to the use of one single buffer for data coming into a storage level as well as for data going out of a storage level. The potential for deadlock of using a common buffer was not discovered during the design of DSH-11 due to the complex interactions of the various protocols for store-behind, read-through, and overflow handling operations.

Section V

DEADLOCK-FREE BUFFER MANAGEMENT SCHEMES

In DSH-11 there are five types of transactions supporting the read-through and store-behind operations. These transactions are : read-through-request (RR), read-through-result (RT), overflow (OV), store-behind-request (SB), and acknowledgment (AK). Each type of transaction is handled differently. Furthermore, the same type of transaction is handled differently depending on whether the transaction is going into or out of a storage level. A potential deadlock exists when different transactions share the same buffer and their paths form a closed loop. We have seen an example of such a deadlock in the PlL3 model where SB transactions coming into a storage level and SB transactions going out of a storage level form a closed loop. Other such potential deadlocks have been discovered in the PlL3 model. This section is focused on developing deadlock-free buffer management algorithms.

Potential deadlocks exist because different transaction types share the same buffer and that the First Come First Served strategy is used for allocating buffer slots. A sim-

ple strategy to avoid deadlock is not to allow buffer sharing among different transaction types. No path crossing can occur thus no loop can exist. Although this strategy is easy to implement, it does not make efficient use of the buffer space. Another strategy to avoid deadlock is to allow buffer sharing, but to make use of more sophisticated buffer allocation algorithms. One such algorithm is discussed below.

5.1 A DEADLOCK-FREE BUFFER ALLOCATION ALGORITHM

Two types of buffers are used at each storage level, the IN buffers and the OUT buffers. Transactions coming into the storage level use the IN buffer and transactions going out of the storage level use the OUT buffer. Transaction coming into a storage level from a higher storage level are the RR, SB, and OV transactions. Transactions coming into a storage level from a lower storage level are the RT and AK stransactions. Similarly, transactions going out of a storage level to the next lower storage level are the RR, SB, and OV transactions. Transactions going out of a storage level to a higher storage level are the RT and AK transactions. Each component in a storage level has an IN buffer and an OUT buffer. This is illustrated in Figure 5.1.

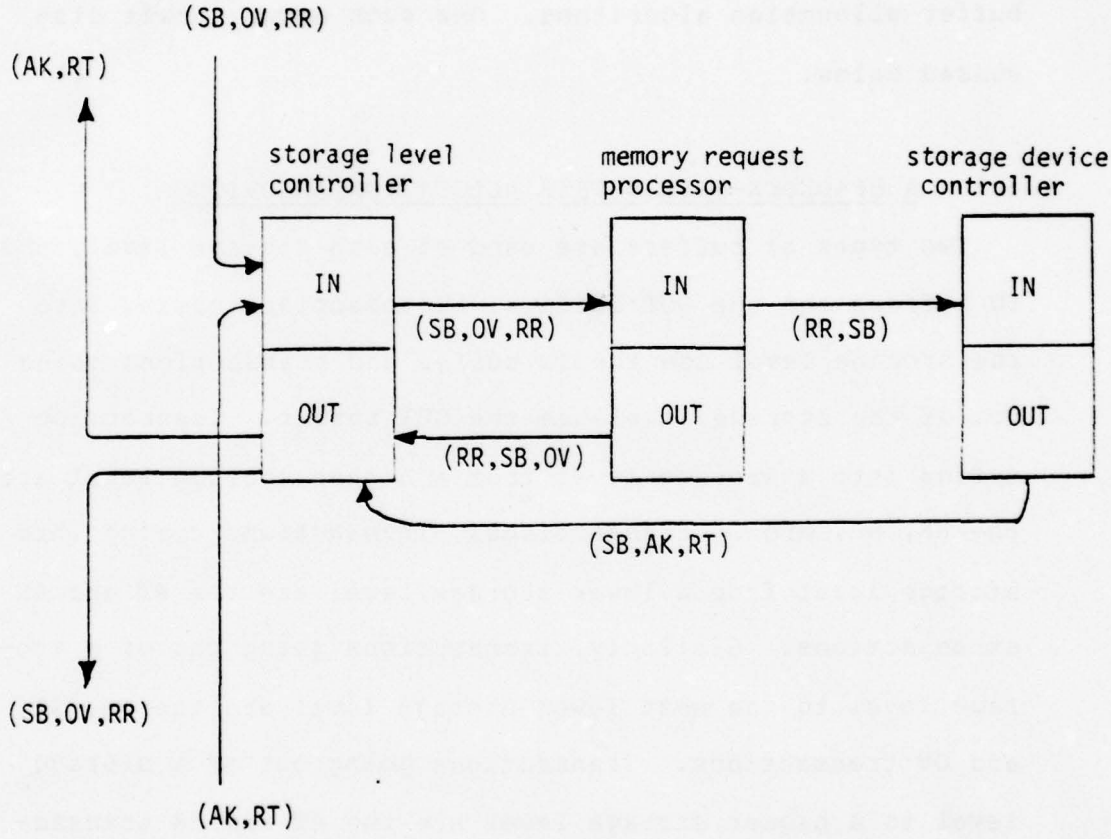

(SB,OV,RR)
(AK,RT)
storage level controller
memory request processor
storage device controller
IN
IN
IN
(SB,OV,RR)
(RR,SB)
OUT
OUT
OUT
(RR,SB,OV)
(SB,AK,RT)
(SB,OV,RR)
(AK,RT)

Figure 5.1

The general idea of this new buffer allocation scheme is not to allow the buffers to become completely filled. When the buffers are filled up to a certain level, only those transactions that can be processed to completion and resulting in freeing up buffer slots are accepted. The precise algorithm is as follows.

1. The size of OUT is always greater than the size of IN.

2. Always maintain at least one empty slot in an IN buffer.

3. The buffer-full (BF) condition is raised when the number of transactions in IN plus the number of transactions in OUT is equal to the size of OUT.

4. If the BF condition exists, then do not accept any RR or SB into a storage level. Only process OV, RT, and AK transactions.

We now provide an informal argument to show that the scheme described above is indeed deadlock-free. First we have to show that the RR and SB transactions are not the only transactions in the system when all the buffer pairs have their BF conditions raised. Then we have to show that processing each of the OV, AK and RT transactions will free up some buffer slots thus lowering some BF conditions.

Suppose that all the BF conditions are raised (i.e., all buffers in a storage level are full). Examine the OUT buffers of the lowest storage level. Since the size of OUT is greater than that of IN, BF implies that there is at least one transaction in OUT. Since this is the lowest storage level, this transaction must be going to a higher storage level, hence cannot be a RR or a SB transaction.

Consider a RT transaction at level i+1 (Figure 5.2).

1. All upper storage levels, level i and level i-1 can receive this transaction since there is always one empty slot in each IN buffer. The departure of the RT transaction creates an empty slot in the OUT buffer of the sender, level i+1.

2. Level i can now send a transaction to level i+1 which creates a slot in level i. The RT transaction can now be serviced in level i.

3. Handling the RT transaction may create an OV transaction in level i. The buffer slot freed up at step 2 is available for the OV transaction in level i.

4. The OV transaction can be sent to level i+1 because there is always a free slot in every IN buffer.

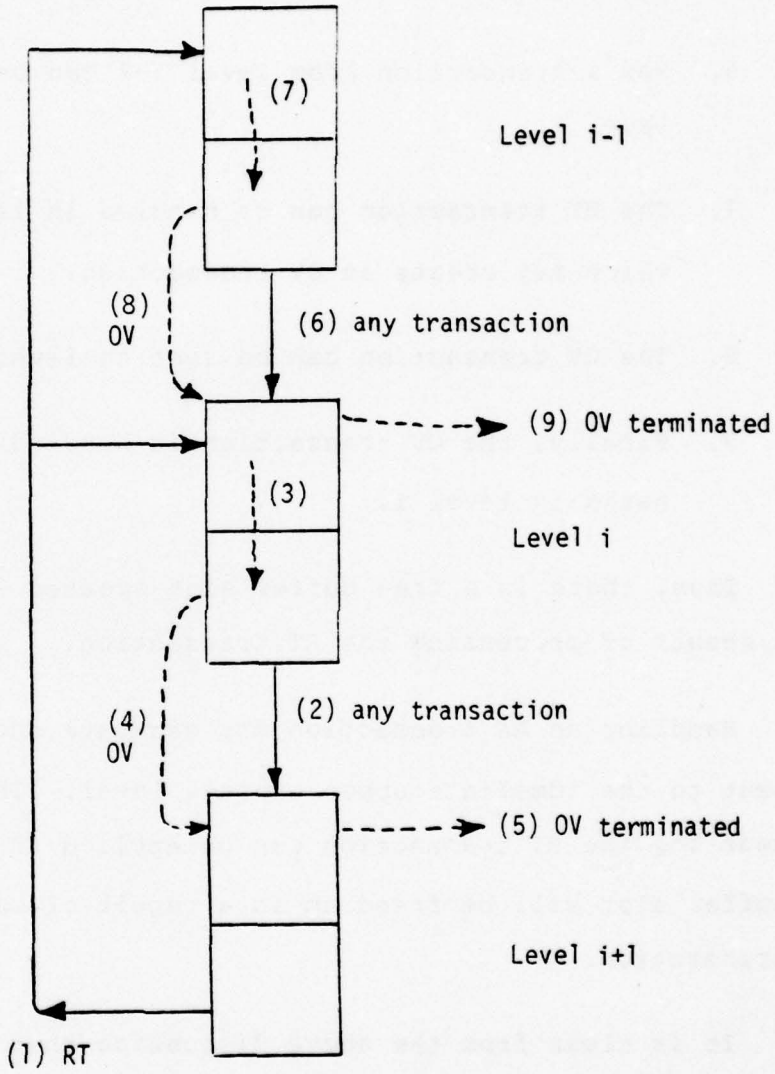

(7)
Level i-1
(8)
OV
(6) any transaction
(9) OV terminated
(3)
Level i
(4)
OV
(2) any transaction
(5) OV terminated
Level i+1
(1) RT

Figure 5.2

5. The OV transaction will be serviced to completion in level i+1. Hence, there is a free slot in level i as result of these operations.

6. Now a transaction from level i-1 can be sent to level i.

7. The RT transaction can be handled in level i-1 which may create an OV transaction.

8. The OV transaction can be sent to level i.

9. Finally, the OV transaction is handled and terminated in level i.

Thus, there is a free buffer slot created in level i-1 as a result of processing the RT transaction.

Handling an AK transaction may generate another AK to be sent to the immediate upper storage level. The same argument for the RT transaction can be applied to show that a buffer slot will be freed up as a result of handling the AK transaction.

It is clear from the above discussion that this buffer management scheme requires more complex protocols among storage levels and a complex priority scheme for the transactions. A key advantage of this scheme is that it makes

efficient use of buffer space since different transactions with varying buffer space requirements can be made to share a common buffer pool.

Section VI

ANOTHER SIMULATION MODEL OF DSH-11 : THE P5L4 MODEL

A GPSS/360 simulation model of another DSH-11 configuration with five processors and four storage levels has been developed. This model is referred to as the P5L4 model. In this model the basic logic used in the P1L3 model was revised to use a deadlock-free buffer management scheme and to accomodate four additional functional hierarchy processors and an additional storage level. For simplicity, the scheme of using separate buffers for different transactions is used for the P5L4 model.

The first series of studies provides further insights to the operation of the store-behind algorithms. It also shows that level 4 storage and its local bus may be too slow to support the amount of data transfer activities at that level.

The second series of studies is aimed at obtaining a better balanced system by increasing the degree of parallelism in the lower storage levels and by reducing the block sizes so as to lower the demand on the buses. A well-balanced system is obtained which is then used as the basic system to

study the effect of using projected 1985 technologies for DSH-11. Results of these studies and their analysis are presented in the following sections, after a brief introduction to the P5L4 model and its parameters.

6.1 THE P5L4 MODEL AND ITS PARAMETERS

The structure of the P5L4 model is very similar to that of the P1L3 model. However, the basic component of the model is quite different. The basic component of the P5L4 model is a facility and a number of data buffers, one for each type of transaction coming into the storage level and going out of the storage level. Figure 6.1(a) illustrates the DSH-11 configuration that P5L4 is modelling, and Figure 6.1(b) illustrates the basic component of the model. A flow chart of the P5L4 model logic is presented in Appendix B. A listing of the P5L4 model is presented in Appendix C.

The parameters used in the P5L4 model are the same as those used in the P1L3 model with the following exceptions. (1) There are five processors, each with 10 degrees of multiprogramming (as opposed to 20 in the P1L3 model). (2) There is a new storage level with 2 storage devices having access times 10 times higher than those of the devices in level 3. The parameters used in the P5L4 model are summarized in Figure 6.2.

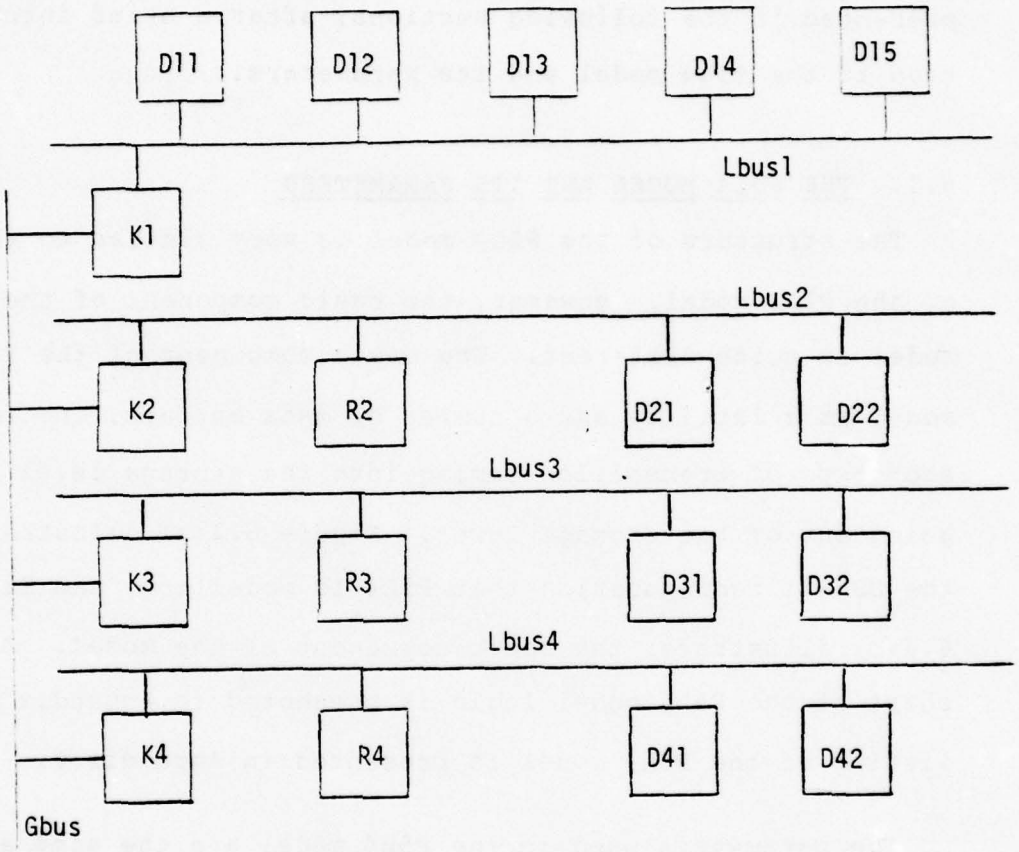


Figure 6.1(a)

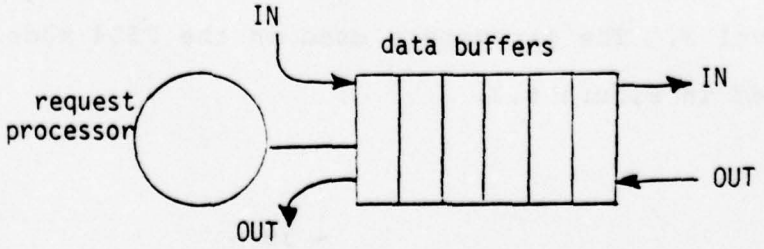


Figure 6.1(b)

DEGREE OF MULTIPROGRAMMING OF A CPU = 10

SIZES OF DATA BUFFERS = 10

DIRECTORY SEARCH TIME = 200 NANOSECONDS

READ/WRITE TIME OF A L(1) STORAGE DEVICE = 100 NS.

READ/WRITE TIME OF A L(2) STORAGE DEVICE = 1000 NS.

READ/WRITE TIME OF A L(3) STORAGE DEVICE = 10000 NS.

READ/WRITE TIME OF A L(4) STORAGE DEVICE = 100000 NS.

BUS SPEED = 10 MHZ

BUS WIDTH = 8 BYTES

SIZE OF A TRANSACTION WITHOUT DATA = 8 BYTES

BLOCK SIZE AT L(1) = 8 BYTES

BLOCK SIZE AT L(2) = 128 BYTES

BLOCK SIZE AT L(3) = 1024 BYTES

BLOCK SIZE AT L(4) = 2048 BYTES

% READ REQUESTS = 70%

% WRITE REQUESTS = 30%

CONDITIONAL PROB. OF FINDING DATA IN A LEVEL
GIVEN THAT THE DATA IS NOT IN ANY UPPER LEVEL = P

FIGURE 6.2

6.2 PRELIMINARY STUDIES USING THE P5L4 MODEL

A preliminary study using the P5L4 model is carried out using several different locality levels and using the parameters listed in Figure 6.2. The simulated time is one millisecond (one million model time units). Results from these studies are summarized in Figure 6.3. Figure 6.3(a) is a table listing the throughput, mean response time, total transaction wait time, total transaction execution time, and 'system utilization'. System utilization is defined as the ratio of the product of the total number of transactions completed and the mean response time to the product of the simulated time and the maximum number of active requests pending at all the processors. It indicates the percentage time that DSH-11 is busy.

Figure 6.3(b) tabulates the utilizations of the buses and the utilizations of typical storage devices at each storage level. The utilizations of all the memory request processors and all the the storage level controllers are very low and are not shown. Figure 6.3(b) shows that the devices and the local bus at level 4 are saturated for all locality levels. The local bus at level 3 is saturated but the devices at level 3 are only 50 percent utilized. Saturation of level 4 at low locality levels is due to the large number of read-through requests that has to be handled at that level.

Locality Level (P)	Throughput (per msec)	Mean Response Time (Nsec)	Total Wait Time (msec)	Total Exec Time (msec)	System Utilization
.50	418	45805	17450	1690	.38
.60	600	35442	19910	1360	.43
.70	717	46664	32490	970	.67
.80	782	43570	32230	840	.68
.95	788	40148	31360	270	.63

FIGURE 6.3(a)

Locality Level (P)	Bus Utilizations					Storage Device Utilization			
	GBUS	LBUS1	LBUS2	BLUS3	LBUS4	L1	L2	L3	L4
.50	.92	.06	.30	.99	.94	.02	.11	.46	.99
.60	.91	.06	.31	.99	.94	.03	.11	.46	.99
.70	.91	.07	.32	.99	.94	.04	.13	.52	.88
.80	.88	.05	.26	.99	.94	.04	.10	.44	.92
.95	.84	.04	.21	.99	.95	.04	.06	.51	.90

FIGURE 6.3(b)

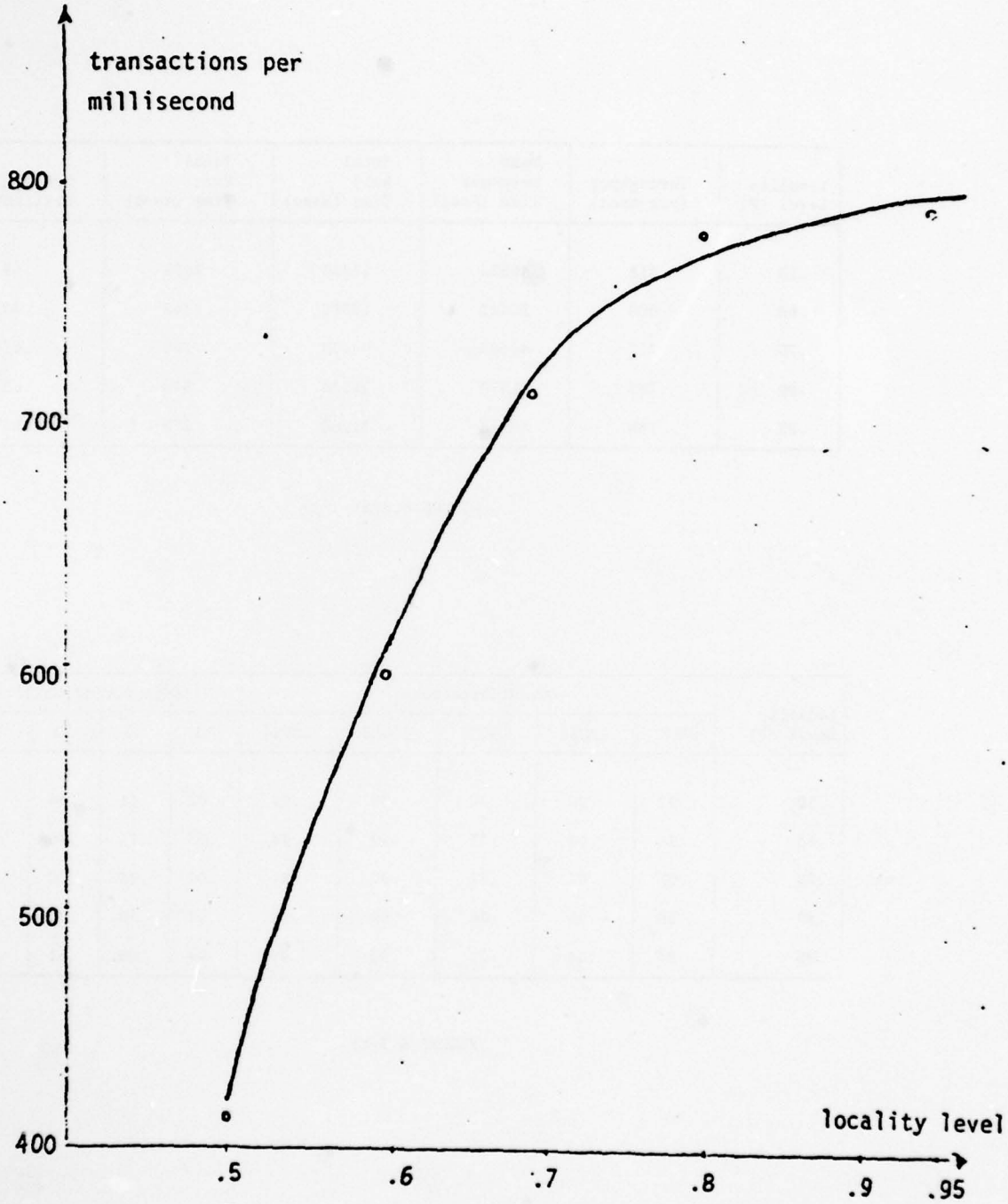

transactions per
millisecond
800
700
600
500
400
locality level
.5
.6
.7
.8
.9
.95

Figure 6.4

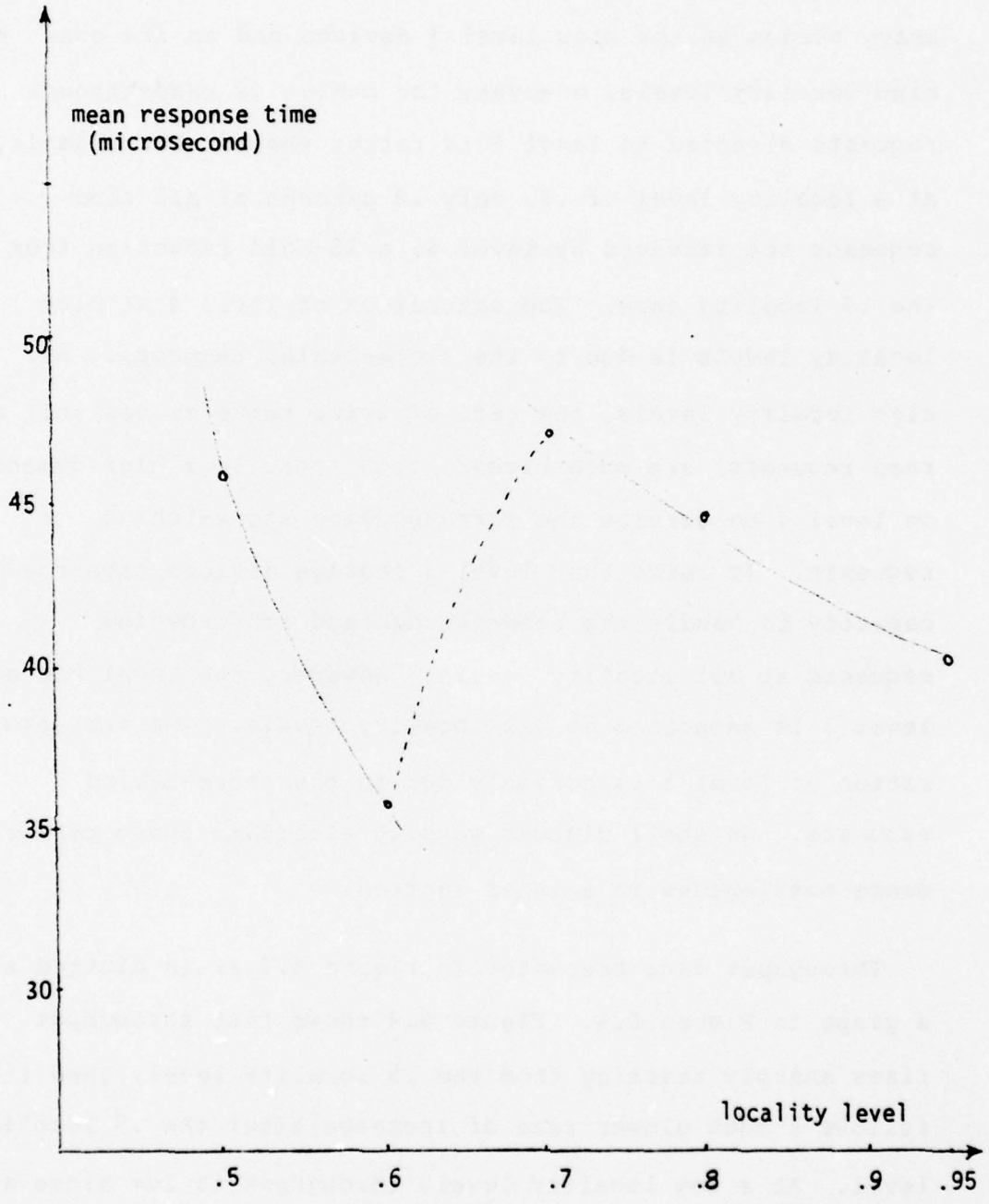

mean response time
(microsecond)
50
45
40
35
30
locality level
.5
.6
.7
.8
.9
.95

Figure 6.5

For example, at a locality level of .5, 12.5 percent of all read requests will be serviced by level 4. This creates a heavy burden on the slow level 4 devices and on its bus. At high locality levels, however, the number of read-through requests directed to level 4 is rather small. For example, at a locality level of .8, only .8 percent of all read requests are serviced by level 4, a 15-fold reduction from the .5 locality case. The saturation of level 4 at high locality levels is due to the store-behind requests. At high locality levels, the rate of write requests (as well as read requests) are much higher, thus there is a high demand on level 4 to service the corresponding store-behind requests. It seems that level 3 storage devices have the capacity to handle the read-through and store-behind requests at all locality levels. However, the local bus at level 3 is saturated at all locality levels. The bus saturation at level 3 is possibly due to the store-behind requests. We shall discuss ways to eliminate these performance bottlenecks in a later section.

Throughput data presented in Figure 6.3(a) is plotted as a graph in Figure 6.4. Figure 6.4 shows that throughput rises sharply starting from the .5 locality level, then its follows a much slower rate of increase after the .7 locality level. At a low locality level, throughput is low since a

large proportion of the read requests has to go to the slower storage devices. As the locality level increases, a large proportion of requests can be handled by the higher storage levels. The higher storage levels are not heavily utilized, thus they can complete the requests quickly. The faster transactions can be completed, the faster new transactions can arrive since the model is a closed one. This explains the sharp increase in throughput between .5 and .7 locality levels.

When the locality level is high, the rate of store-behind transactions coming into the model becomes high. Since there is a fixed proportion of reads and writes in the request stream, the throughput at high locality levels becomes limited by how fast the store-behind requests can be serviced. Thus, at high locality levels, increasing the locality level further will not produce a dramatic increase in throughput.

The plot of mean response time in Figure 6.5 provides further insights to the store-behind operations. Figure 6.5 shows that there is a discontinuity in the mean response time curve between .6 and .7 locality levels. The discontinuity may be explained as follows. As the locality level increases, the rate of store-behind transactions coming into the model also increases. Read operations become a less

dominant factor of system performance. There is a pipeline of buffer slots for store-behind transactions. A write request is completed as soon as it has completed a write to its data cache and has placed a store-behind transaction in the store-behind pipeline. The store-behind transaction flows along the pipeline until it is serviced and terminated by a level 4 storage device. If a write request cannot find a slot in the store-behind pipeline, it has to wait. At high locality levels, the store-behind pipeline becomes full, hence, write operations tend to incure a larger wait time waiting for pipeline slots. It seems that the store-behind pipeline is full after the .7 locality level, causing long wait times by transactions, hence larger mean response times for locality levels higher than .7. The store-behind pipeline is not full for all locality levels below .7. Thus transactions have smaller mean response time in these cases. This expains the difference in behavior of the two mean response time curves.

The data seems to support this theory. Outputs from the simulation runs shows that the pipeline is full for all locality levels greater than and equal to .7. The total transaction time column in Figure 6.3(a) shows that there is a dramatic increase in the transaction wait time for all cases with locality level above .7. The figure also shows

that the transaction wait time is a dominant portion of the total transaction time. Since mean response time is the ratio of total transaction time to total number of completed transactions, the more than doubling of the wait time going from .6 to .7 locality level is the key factor in the sudden increase in mean response time. The sudden increase in wait time is due to the fact that the pipeline is just filled up, new transactions begin to experience prolonged delays. These preliminary studies have provided valuable insights to the dynamics of the store-behind operation. We now have gained enough understanding of the model to tune it for better performance.

6.3 *TUNING THE P5L4 MODEL*

Our objective in this next series of studies is to try to obtain a well-balanced system. From the preliminary studies, we know that to reduce mean response time we have to increase the efficiency of the store-behind pipeline. One approach to increase the efficiency of the pipeline is to increase the parallelism of the lower storage levels, so that the service times of the stages of the pipeline are better balanced. The preliminary studies also reveal that our initial choice of block sizes may not be appropriate for the system.

The approach that is taken to obtain a well-balanced system is as follows. The locality level is fixed at .9. Then the degree of parallelism in level 3 is increased by a factor of 5 and that of level 4 is increased by a factor of 10. Although this would actually be done by increasing the number of devices in these levels, to take advantage of the existing GPSS model, this is accomplished by decreasing the effective service times of the existing devices at these levels appropriately. Finally, the model is run for several choices of block sizes for the storage levels. The results obtained are summarized in Figure 6.6.

The first study uses the same block sizes as those used in the preliminary studies. The results of this study are summarized in column one which clearly shows that level 4 is still the bottleneck causing the very low throughput and high mean response time. Note that none of the storage devices, including level 4, are saturated. This indicates that the bottleneck is caused by the block sizes being too large thus tieing up the bus at level 4 during data transfer.

In the next study, the block sizes between level 2 and level 3 and between level 3 and level 4 are reduced by one half. The results of this study are summarized in column 2. There is significant improvement in throughput and the utilizations of level 4 storage devices, but the bus at level 4 is still a bottleneck.

Block Sizes	Throughput (per msec)	Mean Response Time (Nsec)	Bus Utilization					Storage Device Utilization			
			GBUS	LBUS1	LBUS2	LBUS3	LBUS4	L1	L2	L3	L4
(8,128,1024)	176	258580	.62	.02	.10	.67	1.00	.01	.03	128	.17
(8,64,512)	458	96260	.67	.04	.15	.71	.99	.04	.07	.27	.40
(8,64,256)	721	60940	.77	.07	.26	.84	.99	.06	.11	.28	.83

FIGURE 6.6

Next, the block size between level 3 and level 4 is halved again. This produces a fairly well-balanced system. The results are summarized in column 3. A throughput of .7 million operations per second with mean response time of 61 microseconds is obtained. The utilizations across storage levels are fairly well-balanced.

6.4 COMPARING THE PERFORMANCE OF DSH-11 USING 1979 AND 1985 TECHNOLOGIES

The well-balanced system obtained from the previous studies will be used as a basis for comparing the performance of DSH-11 under 1979 and 1985 technology assumptions. The parameters used in the 1979 case are exactly those used in the well-balanced system of the previous studies. For the 1985 case, we will assume a bus that is 5 times faster than that used in the 1979 case. In general, the speeds of the storage devices in the 1985 case will be faster. We estimate that the level 1 storage devices will be twice as fast in 1985 as in 1979. All other devices are estimated to be 10 times faster in 1985 than in 1979. Lastly, we expect 1985 to produce better associative processors for directory searching thus the directory search time will be reduced by one half in 1985. These estimates will be incorproated in the parameters for the 1985 case.

The model using 1979 technology assumptions is run for 4 different request streams with different proportions of reads and writes. The model using 1985 technology assumptions is then run with the same 4 different request streams. A locality level of .9 was used in all these cases. The results are summarized in Figure 6.7.

The throughputs for the two cases are plotted on the same graph in Figure 6.8. In general, for both cases, throughput increases as the proportion of read requests increases. It can be inferred from the results that the throughput of DSH-11 using 1985 technology is between 5 to 10 times better than using 1979 technology. For a request stream with 70 percent read requests and 30 percent write requests, DSH-11 using 1979 technology can support a throughput of .7 million requests per second with a mean response time of 61 microseconds. For the same mix of requests, DSH-11 using 1985 technology can support a throughput of 4 million requests per second with a mean response time of 10 microseconds.

% Read	Throughput (per msec)	Mean Response Time (Nsec)	Bus Utilizations					Storage Utilizations			
			GBUS	LBUS1	LBUS2	LBUS3	LBUS4	L1	L2	L3	L4
.50	450	97580	.76	.06	.25	.84	.99	.04	.10	.25	.67
.70	721	60940	.77	.07	.26	.84	.99	.06	.11	.28	.65
.80	1559	26790	.85	.10	.34	.91	.97	.11	.18	.34	.71
.90	3239	13440	.90	.14	.42	.93	.97	.23	.28	.35	.83

(1979 Technology)

% Read	Throughput (per msec)	Mean Response Time (Nsec)	Bus Utilizations					Storage Utilizations			
			GBUS	LBUS1	LBUS2	LBUS3	LBUS4	L1	L2	L3	L4
.50	2298	19780	.76	.06	.24	.82	.99	.13	.05	.27	.35
.70	4320	9940	.79	.07	.28	.86	.98	.20	.06	.28	.34
.80	15040	2640	.96	.15	.47	.97	.92	.64	.14	.38	.28
.90	22760	1760	.95	.16	.47	.96	.91	.99	.17	.27	.34

(1985 Technology)

FIGURE 6.7

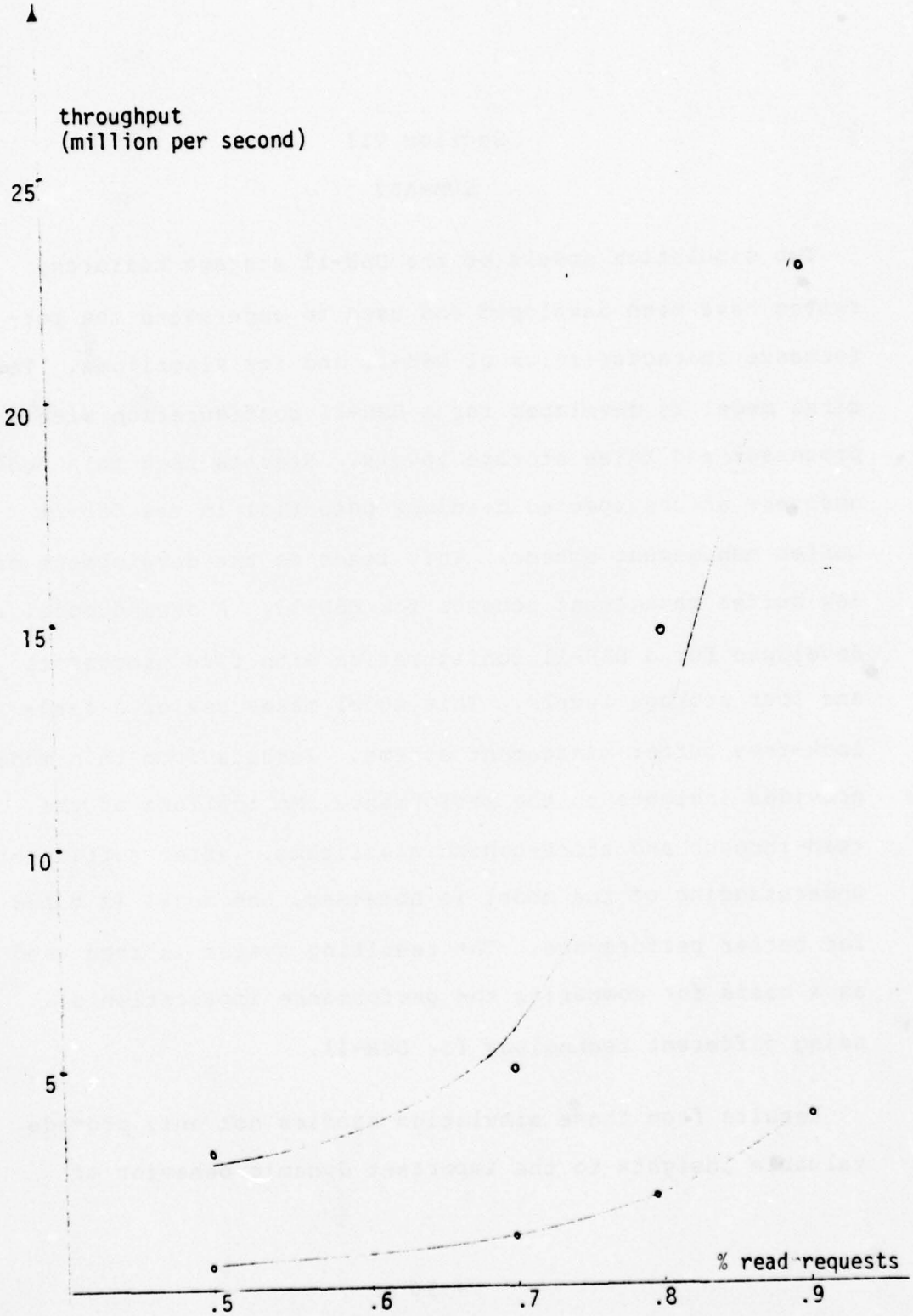

throughput
(million per second)
25
20
15
10
5
% read requests
.5
.6
.7
.8
.9

Figure 6.8

Section VII

SUMMARY

Two simulation models of the DSH-11 storage hierarchy system have been developed and used to understand the performance characteristics of DSH-11 and its algorithms. The first model is developed for a DSH-11 configuration with one processor and three storage levels. Results from this model uncovers an unsuspected deadlock potential in the DSH-11 buffer management scheme. This leads to the development of new buffer management schemes for DSH-11. A second model is developed for a DSH-11 configuration with five processors and four storage levels. This model makes use of a deadlock-free buffer management scheme. Results from this model provides insights to the performance implications of the read-through and store-behind algorithms. After sufficient understanding of the model is obtained, the model is tuned for better performance. The resulting system is then used as a basis for comparing the performance implication of using different technology for DSH-11.

Results from these simulation studies not only provide valuable insights to the important dynamic behavior of

store-behind and read-through algorithms, they also provide indications that the DSH-11 is capable of supporting the menory requirements of the IMS functional hierarchy.

REFERENCES

(Denning, 1970): Denning, P.J., 'Virtual Memory', ACM Computing Surveys, 2, 3 (September 1970), 153-190.

(Lam and Madnick, 1979a): Lam, C.Y., and Madnick, S.E., 'Technical Report No. 1: The Intelligent Memory System Architecture - Research Directions', M.I.T. Sloan School Internal Report No. M010-7908-01, August 1979.

(Lam and Madnick, 1979b): Lam, C.Y., and Madnick, S.E., 'Technical Report No. 2: The IMS Data Storage Hierarchy - DSH-1', M.I.T. Sloan School Internal Report No. M010-7908-02, August 1979.

(Lam and Madnick, 1979c): Lam, C.Y., and Madnick, S.E., 'Technical Report No. 3: The IMS Data Storage Hierarchy - DSH-11', M.I.T. Sloan School Internal Report No. M010-7908-03, August 1979.

Appendix A

LISTING OF THE P1L3 MODEL

```
//LAM1 JOB LAM,MPROFILE='RETURN',
// PROFILE='HIGH',
// TIME=2
//*PASSWORD
//GPSS PROC
//C   EXEC  PGM=DAG01,TIME=2
//STEPLIB DD DSN=POTLUCK.LIBRARY.GPSS.LOAD,DISP=SHR
//DOUTPUT DD SYSOUT=PROFILE=RETURN,DCB=BLKSIZE=931
//DINTERO DD UNIT=SCRATCH,SPACE=(CYL,(1,1)),DCB=BLKSIZE=1880
//DSYMTAB DD UNIT=SCRATCH,SPACE=(CYL,(1,1)),DCB=BLKSIZE=7112
//DREPTGEN DD UNIT=SCRATCH,SPACE=(CYL,(1,1)),DCB=BLKSIZE=800
//DINTWORK DD UNIT=SCRATCH,SPACE=(CYL,(1,1)),DCB=BLKSIZE=2680
// PEND
//STEP1  EXEC  GPSS,PARM=C
//DINPUT1  DD *
*****************************************************************
*                                                               *
*     TRANSACTION PARAMETER USAGE                               *
*                                                               *
*     P1      CPU IDENTIFIER                                    *
*     P2      ARRIVAL TIME                                      *
*     P3      COMPLETION TIME                                   *
*     P4      TOTAL EXECUTION TIME                              *
*     P5      TOTAL ELAPSED TIME                                *
*     P6      TOTAL WAIT TIME                                   *
*     P7      SERVICE TIME                                      *
*     P11     DUMMY                                             *
*                                                               *
*****************************************************************
*
 NTXN   EQU        01,X        NUMBER OF TXNS PROCESSED
 SUMX   EQU        02,X        EXECUTION TIME OF ALL TXNS
 SUMQ   EQU        03,X        QUEUE TIME OF ALL TXNS
 SUMW   EQU        04,X        ELAPSED TIME OF ALL TXNS
*
 MAXMP  EQU        05,X        DEGREE OF CPU MULTIPLROGRAMMING
 NREAD  EQU        06,X        PARTS IN THOUSAND OF READ TXNS
 NWRIT  EQU        07,X        PARTS IN THOUSAND OF WRITE TXNS
*
 PIN1   EQU        08,X        PROB OF FINDING READ DATA IN L(1)
 PIN2   EQU        09,X        PROB OF FINDING READ DATA IN L(2)
 PIN3   EQU        10,X        PROB OF FINDING READ DATA IN L(3)
*
 POV1   EQU        11,X        PROB OF OVERFLOW FROM L(1)
 POV2   EQU        12,X        PROB OF OVERFLOW FROM L(2)
 POV3   EQU        13,X        PROB OF OVERFLOW FROM L(3)
*
*****************************************************************
*        MAXIMUM DATA QUEUE LENGTHS                             *
*****************************************************************
*
 DXM11  EQU        14,X
 DYM11  EQU        15,X
 DXM12  EQU        16,X
 DYM12  EQU        17,X
```

```
 DXM13 EQU      18,X
 DYM13 EQU      19,X
*
 DXM21 EQU      20,X
 DYM21 EQU      21,X
 DXM22 EQU      22,X
 DYM22 EQU      23,X
*
 DXM31 EQU      24,X
 DYM31 EQU      25,X
 DXM32 EQU      26,X
 DYM32 EQU      27,X
*
 KXM1  EQU      28,X
 KYM1  EQU      29,X
*
 KXM2  EQU      30,X
 KYM2  EQU      31,X
*
 KXM3  EQU      32,X
 KYM3  EQU      33,X
*
 RXM2  EQU      34,X
 RYM2  EQU      35,X
*
 RXM3  EQU      36,X
 RYM3  EQU      37,X
*
*************************************************************
*     CURRENT LENGTHS OF DATA QUEUES                        *
*************************************************************
*
 DXL11 EQU      38,X
 DYL11 EQU      39,X
 DXL12 EQU      40,X
 DYL12 EQU      41,X
 DXL13 EQU      42,X
 DYL13 EQU      43,X
*
 DXL21 EQU      44,X
 DYL21 EQU      45,X
 DXL22 EQU      46,X
 DYL22 EQU      47,X
*
 DXL31 EQU      48,X
 DYL31 EQU      49,X
 DXL32 EQU      50,X
 DYL32 EQU      51,X
*
 KXL1  EQU      52,X
 KYL1  EQU      53,X
*
 KXL2  EQU      54,X
 KYL2  EQU      55,X
*
```

```
KXL3   EQU      56,X
KYL3   EQU      57,X

RXL2   EQU      58,X
RYL2   EQU      59,X
*
RXL3   EQU      60,X
RYL3   EQU      61,X
*
*************************************************************
* SERVICE TIMES OF DEVICES, BUSES, PROCESSORS               *
*************************************************************

DEX11  EQU      62,X        L(1) STORAGE SERVICE TIME
DEX12  EQU      63,X
DEX13  EQU      64,X
DEX21  EQU      65,X        L(2) STORAGE SERVICE TIMES
DEX22  EQU      66,X
DEX31  EQU      67,X        L(3) STORAGE SERVICE TIMES
DEX32  EQU      68,X
BEXD1  EQU      69,X        BUS SERV TIME L(1)
BEXD2  EQU      70,X        BUS SERV. TIME L(2)
BXD3   EQU      71,X        BUS SERV. TIME L(3)
BEXM   EQU      72,X        BUS SERV. TIME FOR MSG
KEX    EQU      73,X        LEVEL CONTROLLER (K) SERVICE TIME
PEX    EQU      74,X        MEMORY REQUEST PROCESSOR (R) SERVICE TIME
TIMER  EQU      75,X

*************************************************************
*                                                           *
*  VARAIBLE DEFINITIONS                                     *
*                                                           *
*************************************************************

MRESP  FVARIABLE  (X$SUMW/X$NTXN)          MEAN RESPONSE TIME
TXNW   VARIABLE   P3-P2                    TXN ELAPSED TIME
TXNQ   VARIABLE   P3-P2-P4                 TXN WAIT TIME
TXNX   VARIABLE   P4
RTOK   BVARIABLE  (X$KXL1'L'X$KXM1)*(X$KXL2'L'X$KXM2)*FNU$GBUS
BVA1   BVARIABLE  (X$DYL11'L'X$DYM11)*FNU$DRP11
BVA2   BVARIABLE  (X$KXL1'L'X$KXM1)*FNU$LBUS1
BVA3   BVARIABLE  (X$DYL21'L'X$DYM21)*FNU$DRP21
BVA21  BVARIABLE  (X$DYL22'L'X$DYM22)*FNU$DRP22
BVA4   BVARIABLE  (X$KXL2'L'X$KXM2)*FNU$LBUS2
BVA5   BVARIABLE  (X$KYL2'L'X$KYM2)*FNU$KRP2
BVA6   BVARIABLE  (X$KXL1'L'X$KXM1)*FNU$GBUS
BVA7   BVARIABLE  (X$DXL11'L'X$DXM11)*FNU$LBUS1
BVA8   BVARIABLE  (X$DYL31'L'X$DYM31)*FNU$DRP31
BVA22  BVARIABLE  (X$DYL32'L'X$DYM32)*FNU$DRP32
BVA9   BVARIABLE  (X$KXL3'L'X$KXM3)*FNU$LBUS3
BVA10  BVARIABLE  (X$KYL3'L'X$KYM3)*FNU$KRP3
BVA11  BVARIABLE  (X$PXL2'L'X$PXM2)*FNU$LBUS2
BVA12  BVARIABLE  (X$RYL2'L'X$RYM2)*FNU$PRP2
BVA13  BVARIABLE  (X$DXL21'L'X$DXM21)*FNU$LBUS2
BVA23  BVARIABLE  (X$DXL22'L'X$DXM22)*FNU$LBUS2
```

```
 BVA14 BVARIABLE   (X$KYL1*L'X$KYM1)*FNU$KRP1
 BVA15 BVARIABLE   (X$KXL2*L'X$KXM2)*FNU$GBUS
 BVA16 BVARIABLE   (X$KXL3*L'X$KXM3)*FNU$GBUS
 BVA17 BVARIABLE   (X$PXL3*L'X$PXM3)*FNU$LBUS3
 BVA18 BVARIABLE   (X$RYL3*L'X$RYM3)*FNU$RRP3
 BVA19 BVARIABLE   (X$DXL31*L'X$DXM31)*FNU$LBUS3
 BVA24 BVARIABLE   (X$DXL32*L'X$DXM32)*FNU$LBUS3
 BVA20 BVARIABLE   (X$KYL1*L'X$KYM1)*FNU$KRP1

*****************************************************************
*                                                               *
*  QTABLE DEFINITIONS - DISTRIBUTIONS OF QUEUE LENGTHS          *
*                                                               *
*****************************************************************

*****************************************************************
*                                                               *
* FUNCTION DEFINITIONS                                          *
*                                                               *
*****************************************************************

 WICHW FUNCTION   P1,D3
2,WWW11/3,WWW12/4,WWW13

 WICHA FUNCTION   P1,D3
2,AAA11/3,AAA12/4,AAA13


*****************************************************************
*                                                               *
* TABLE DEFINITIONS - DISTRIBUTIONS OF TXN ELAPSED TIME,        *
*                     WAIT TIME                                 *
*                                                               *
*****************************************************************

 TXNW  TABLE      V$TXNW,100,100,100
 TXNQ  TABLE      V$TXNQ,100,100,100
 TXNX  TABLE      V$TXNX,100,100,100


*****************************************************************
*                                                               *
*  INITIALIZE CONSTANTS                                         *
*                                                               *
*****************************************************************

       INITIAL    X$MAXMP,20        DEGREE OF MULTIPROGRAMMING OF A CPU
       INITIAL    X$NREAD,700       % READ TXN
       INITIAL    X$NWRIT,300       % WRITE TXN
       INITIAL    X$PIN1,400        PROB OF FINDING READ DATA IN L(1)
       INITIAL    X$PIN2,400        PROB OF NOT IN L(1) AND IN L(2)
       INITIAL    X$PIN3,1000       PROB OF FINDING DATA IN L(3)
       INITIAL    X$POV1,500        PROB OF OVERFLOW FROM L(1)
       INITIAL    X$POV2,500        PROB OF OVERFLOW FROM L(2)
       INITIAL    X$DXM11,10        MAXIMUM DATA QUEUE LENGTH
```

```
        INITIAL    X$DYM11,10
        INITIAL    X$DXM12,10
        INITIAL    X$DYM12,10
        INITIAL    X$DXM13,10
        INITIAL    X$DYM13,10
        INITIAL    X$DXM21,10
        INITIAL    X$DYM21,10
        INITIAL    X$DXM22,10
        INITIAL    X$DYM22,10
        INITIAL    X$DXM31,10
        INITIAL    X$DYM31,10
        INITIAL    X$DXM32,10
        INITIAL    X$DYM32,10
        INITIAL    X$KXM1,10
        INITIAL    X$KYM1,10
        INITIAL    X$KXM2,10
        INITIAL    X$KYM2,10
        INITIAL    X$KXM3,10
        INITIAL    X$KYM3,10
        INITIAL    X$RXM2,10
        INITIAL    X$RYM2,10
        INITIAL    X$RXM3,10
        INITIAL    X$RYM3,10
        INITIAL    X$DEX11,100      ACCESS TIME OF D11 IN NANOSEC
        INITIAL    X$DEX12,100
        INITIAL    X$DEX13,100
        INITIAL    X$DEX21,1000     ACCESS TIME OF D21 IN NANOSEC
        INITIAL    X$DEX22,1000
        INITIAL    X$DEX31,10000    ACCESS TIME OF D31 IN NANOSEC
        INITIAL    X$DEX32,10000
        INITIAL    X$BEXD1,100       BUS SERV. TIME IN NANOSEC
        INITIAL    X$BEXD2,1600
        INITIAL    X$DEXM,100
        INITIAL    X$KEX,100         L(I) CONTR. P. SERV. TIME IN NANOS
        INITIAL    X$REX,200         REQ. P. SERVICE TIME IN NANOS
        INITIAL    X$TIMER,1000000   SIMULATION TIME


**************************************
*                                    *
* MACRO -UTX                         *
*                                    *
**************************************

 UTX    STARTMACRO
        SEIZE      #A
        DEPART     #B
        ASSIGN     4+,#C
        ASSIGN     7,#C
        ADVANCE    P7
        RELEASE    #A
        ENDMACRO

**************************************
```

```
*                                    *
*   MACRO - UQTQ                     *
*                                    *
**************************************

 UQTQ  STARTMACRO
       QUEUE       #A
       SEIZE       #B
       DEPART      #A
       ASSIGN      4+,#D
       ASSIGN      7,#D
       ADVANCE     P7
       RELEASE     #B
       QUEUE       #C
       ENDMACRO

*************************
*                       *
*  MACRO - UQT          *
*                       *
*************************

 UQT   STARTMACRO
       QUEUE       #A
       SEIZE       #B
       DEPART      #A
       ASSIGN      4+,#C
       ASSIGN      7,#C
       ADVANCE     P7
       RELEASE     #B
       ENDMACRO

************************************
*                                  *
*  MACRO - UQDQ                    *
*                                  *
************************************

 UQDQ  STARTMACRO
       QUEUE       #A
       TEST E      #G,1
       SAVEVALUE   #D,1
       SEIZE       #E
       DEPART      #A
       SAVEVALUE   #D,1
       ASSIGN      4+,#7
       ASSIGN      7,#F
       ADVANCE     P7
       RELEASE     #E
       QUEUE       #C
       ENDMACRO


*************************
*                       *
```

```
*  MACRO - UQD          *
*                       *
*************************

 UQD   STARTMACRO
       QUEUE       #A
       TEST E      #G,1
       SAVEVALUE   #D,1
       SEIZE       #E
       DEPART      #A
       SAVEVALUE   #B,1
       ASSIGN      4+,#F
       ASSIGN      7,#F
       ADVANCE     P7
       RELEASE     #E
       ENDMACRO

************************************
*                                  *
*   MACRO - FINI                   *
*                                  *
************************************

 FINI  STARTMACRO
       MARK        3
       SAVEVALUE   NTXN+,1
       SAVEVALUE   SUMX+,V$TXNX
       SAVEVALUE   SUMQ+,V$TXNQ
       SAVEVALUE   SUMW+,V$TXNW
       SAVEVALUE   MRESP,V$MRESP
       TABULATE    TXNW
       TABULATE    TXNQ
       TABULATE    TXNX
       ASSIGN      1,0
       ASSIGN      2,0
       ASSIGN      3,0
       ASSIGN      4,0
       ASSIGN      5,0
       ASSIGN      6,0
       ENDMACRO

       SIMULATE
************************************
*                                  *
*   CPU #1                         *
*                                  *
************************************

 CPU1  GENERATE    ,,,XSMAXMP,,,F
*************************
* START FOR CPU1 TXNS   *
*************************
 STAR1 PRIORITY    9                SET HIGH PRIORITY
       MARK        2                ARRIVAL TIME OF TXN
       ASSIGN      1,1              SET CPU ID = 1
```

```
       TRANSFER   .X$NREAD,WWW1,RER1     READ OR WRITE TXN?
**************************
* READ TXN FROM CPU1     *
**************************
 RER1  QUEUE      DIQ11                 READ TXN
       SEIZE      DRP11
       DEPART     DIQ11
       PRIORITY   0                     RESET PRIORITY
       ASSIGN     4+,X$REX              TIME FOR DIRECTORY SEARCH
       ASSIGN     7,X$REX
       ADVANCE    P7
       RELEASE    DRP11
       TRANSFER   .X$PIN1,NIN11,IND11 IS DATA IN L(1)?
**************************
* READ TXN FROM CPU1     *
* IS SATISFIED IN L(1)   *
**************************

 IND11 ASSIGN     11,0

**************************
* READ DATA FROM D11     *
**************************

 UQT   MACRO      DIQ11,DRP11,X$DEX11

**************************
* USE FINI MACRO         *
* THE TXN IS COMPLETED   *
**************************

 FINI  MACRO
       TRANSFER   ,STAR1               THE TXN BECOMES A NEW TXN
**************************
* READ TXN FROM CPU1 IS  *
* NOT SATISFIED IN L(1)  *
**************************

 NIN11 QUEUE      DOQ11

**************************
* USE UTX TO USE         *
* THE LOCAL BUS LBUS1    *
**************************
 UTX   MACRO      LBUS1,DOQ11,X$BEXN
       TRANSFER   ,COMR                GO TO COMMON CODE FOR READ
**************************
* WRITE TXN FROM CPU1    *
**************************
 WWW1  QUEUE      DIQ11
       TEST E     BV$BVA1,1            D11 OUT QUEUE AND DRP FREE?
       SAVEVALUE  DYL11+,1             SAVE SPACE IN OUT Q
       SEIZE      DRP11
       PRIORITY   0                     RESET PRIORITY
       DEPART     DIQ11
```

```
        ASSIGN      4+,X$DEX11        TIME FOR WRITING DATA
        ASSIGN      7,X$DEX11
        ADVANCE     P7
        RELEASE     DRP11
        SPLIT       1,STB1             CREATE A STORE-BEHIND TXN
*************************
* WRITE TXN IS COMPLETED*
*************************
 FINI   MACRO
        TRANSFER    ,STAR1             BECOMES A NEW TXN FROM CPU1
*************************
* STORE-BEHIND TXN      *
*************************
 STB1   QUEUE       DYQ11               PUT TXN IN DATA QUEUE

        TEST E      BV$BVA2,1          K1 IN-Q AND LBUS1 FREE ?
        SAVEVALUE   KXL1+,1            RESERVE SPACE IN IN-Q

*************************
* USE LBUS1 TO SEND TXN   *
* FROM D11 TO K1          *
*************************
 UTX    MACRO       LBUS1,DYQ11,X$BEXD1
        SAVEVALUE   DYL11-,1            RELEASE SPACE IN D11
        TRANSFER    ,COMW               TO COMMON CODE FOR WRITE

*************************
* COMMON CODE FOR       *
* READ TO LOWER LEVELS  *
* JOINED BY ALL CPUS    *
*************************

 COMR   ASSIGN      11,0                DUMMY STATEMENT
*************************
* USE K1                *
*************************
 UQTO   MACRO       KIQ1,KRP1,KOQ1,X$KEX
*************************
* USE GLOBAL BUS GBUS   *
*************************
 UTX    MACRO       GBUS,KOQ1,X$BEXM
*************************
* USE K2                *
*************************
 UQTO   MACRO       KIQ2,KRP2,KOQ2,X$KEX
*************************
*  USE LOCAL BUS LBUS2    *
*************************
 UTX    MACRO       LBUS2,KOQ2,X$BEXM
*************************
* USE R2 TO SEE IF DATA *
* IS IN L(2)            *
*************************

 UQT    MACRO       RIQ2,RRP2,X$REX
```

```
       TRANSFER   .XSPIN2,NIN2,INL2     IS DATA IN L(2)?
**************************
* DATA IS NOT FOUND IN   *
* L(2)                   *
**************************

 NIN2  QUEUE      ROQ2

**************************
* USE LBUS2 SEND TXN TO  *
* K2                     *
**************************
 UTX   MACRO      LBUS2,ROQ2,X$BEXM
**************************
*  SERVICED BY K2        *
**************************
 UQTO  MACRO      KIQ2,KRP2,KOQ2,X$KEX
**************************
* USE GBUS SEND TXN  TO  *
* K3                     *
**************************
 UTX   MACRO      GBUS,KOQ2,X$BEXM
**************************
* SERVICED BY K3         *
**************************
 UQTO  MACRO      KIQ3,KRP3,KOQ3,X$KEX
**************************
* USE LBUS3 SEND TXN TO  *
* R3                     *
**************************
 UTX   MACRO      LBUS3,KOQ3,X$BEXM
**************************
* SEARCH DIRECTORY IN    *
* R3 FOR DATA            *
**************************

 UQT   MACRO      RIQ3,RRP3,X$REX
       TRANSFER   ,INL3            DATA IS IN L(3)
************************************
* DATA IS FOUND IN L(2), READ THE  *
* DATA AND SEND IT UP TO L(1)      *
************************************

 INL2  QUEUE      ROQ2

**************************
* SEND TXN TO DEVICE     *
* VIA LBUS2              *
**************************
 UTX   MACRO      LBUS2,ROQ2,X$BEXM

************************************
* IS DATA IN D11 OR D12?           *
************************************
```

```
        TRANSFER   .5,RRR21,RRR22

**********************
* DATA IS IN D11     *
**********************

RRR21   QUEUE      DIQ21              QUEUE TO RETRIEVE DATA
        TEST E     BV$BVA3,1          D21 OUT-Q AND DRP21 FREE?
        SAVEVALUE  DYL21+,1           SAVE SPACE IN D21 OUT-Q

**************************
* USE D21 TO RETRIEVE    *
* THE DATA               *
**************************
UTX     MACRO      DRP21,DIQ21,X$DEX21  RETRIEVE THE DATA
        QUEUE      DYQ21                PUT DATA IN SLOT

        TEST E     BV$BVA4,1          K2 IN-Q AND LBUS2 FREE?
        SAVEVALUE  KXL2+,1            RESERVE K2 IN-Q SLOT

**************************
* USE LBUS2 SEND DATA TO *
* K2                     *
**************************
UTX     MACRO      LBUS2,DYQ21,X$BEXD1

        SAVEVALUE  DYL21-,1           RELEASE SLOT IN D21 OUT-QUEUE
        TRANSFER   ,RTF2              TO CODE FOR READ-THROUGH FROM L(2)

**********************
* DATA IS IN D22     *
**********************

RRR22   QUEUE      DIQ22
        TEST E     BV$BVA21,1
        SAVEVALUE  DYL22+,1

UTX     MACRO      DRP22,DIQ22,X$DEX22

        QUEUE      DYQ22
        TEST E     BV$BVA4,1
        SAVEVALUE  KXL2+,1

UTX     MACRO      LBUS2,DYQ22,X$BEXD1

        SAVEVALUE  DYL22-,1
        TRANSFER   ,RTF2

************************************
* READ THROUGH FROM LEVEL L(2)     *
************************************

RTF2    ASSIGN     11,0
```

```
*******************************
* SERVICED BY K2              *
*******************************
 UQDQ  MACRO     KXQ2,KXL2-,KYQ2,KYL2+,KRP2,X$KEX,BV$BVA5

       TEST E    BV$BVA6,1          K1 IN-Q AND GBUS FREE?
       SAVEVALUE KXL1+,1            RESERVE K1 IN-Q SLOT

************************
* USE GBUS TO SEND DATA TO*
* K1                   *
************************
 UTX   MACRO     GBUS,KYQ2,X$BEXD1
       SAVEVALUE KYL2-,1            RELEASE SLOT IN K2

************************************
* STORE DATA INTO L(1) AS A RESULT*
* OF READ-THROUGH                  *
************************************

 STOR1 ASSIGN    11,0

************************
*  SERVICED BY K1      *
************************

 UQD   MACRO     KXQ1,KXL1-,,KYL1+,KRP1,X$KEX,BV$BVA20


*********************************
* SEND TO D11 OR D12            *
*********************************

       SPLIT     1,FN$WICHW,1       WHICH DATA CACHE TO GO?
       TERMINATE

*********************
* STORE TO D11      *
*********************

 WWW11 ASSIGN    11,0               WRITE TO D11
       QUEUE     KYQ1
       TEST E    BV$BVA7,1          SPACE IN D11 IN-Q AND LBUS1 FREE?
       SAVEVALUE DXL11+,1           YES, RESERVE A SLOT

************************
* SEND TXN TO D11 VIA  *
* LBUS1                *
************************

 UTX   MACRO     LBUS1,KYQ1,X$BEXD1

       SAVEVALUE KYL1-,1            RELEASE K1 SLOT

************************
```

```
*  WRITE DATA TO D11     *
*************************

 UQT   MACRO       DXQ11,DRP11,XSDEX11

       SAVEVALUE   DXL11-,1
       TRANSFER    .XSPOV1,NOV11,OVL11     ANY OVERFLOW FROM L(1)?

*************************
* NO OVERFLOW FROM L(1) *
*************************

 NOV11 ASSIGN      11,0

*************************
* THE READ TXN HAS ENDED*
*************************

 FINI  MACRO
       TRANSFER    ,STAR1

*************************
* THERE IS OVERFLOW FROM*
* L(1), END THE READ    *
* TXN, AT THE SAME TIME *
* HANDLE THE OVERFLOW   *
*************************

 OVL11 SPLIT       1,OVF11            GOT OVERFLOW HANDLING
 FINI  MACRO                         AT THE SAME TIME END THE TXN
       TRANSFER    ,STAR1

*************************
* OVERFLOW HANDLING FOR *
* D11                   *
*************************

 OVF11 ASSIGN      11,0

 UQT   MACRO       DOQ11,LBUS1,XSBEXM
       TRANSFER    ,OVL1              GOTO COMMON CODE FOR OVERFLOW

*************************************
*  WWW12                            *
*************************************

*************************************
*   WWW13                           *
*************************************

 WWW12 ASSIGN      11,0
 WWW13 ASSIGN      11,0
*************************************
* COMMON CODE FOR OVERFLOW FROM     *
* L(1)                              *
```

```
**********************************

OVL1   ASSIGN     11,C

****************************************
* USE  K1, THEN GBUS, THEN K2   *
* THEN LBUS2, THEN USE R2              *
****************************************

 UQTQ  MACRO      KIQ1,KRP1,KOQ1,X$KEX

 UTX   MACRO      GBUS,KOQ1,X$BEXM

 UQTQ  MACRO      KIQ2,KRP2,KOQ2,X$KEX

 UTX   MACRO      LBUS2,KOQ2,X$BEXM

       QUEUE      RIQ2

 UTX   MACRO      RRP2,RIQ2,X$REX

       TERMINATE

*************************
* DATA IS FOUND IN L(3) *
*************************

 INL3  QUEUE      ROQ3

*************************
* USE LBUS3 SEND TXN TO   *
* D31                  *
*************************

 UTX   MACRO      LBUS3,ROQ3,X$BEXM

******************************
* READ FROM D31 OR D32?     *
******************************

       TRANSFER   .5,RRR31,RRR32

********************
* READ FROM D31    *
********************

 RRR31 QUEUE      DIQ31
       TEST E     BV$BVA8,1        SPACE IN D31 OUT-Q AND DRP31 FREE?
       SAVEVALUE  DYL31+,1

*************************
* READ DATA FROM D31    *
*************************
```

```
 UTX   MACRO     DRP31,DIQ31,X$DEX31

       QUEUE     DYQ31
       TEST E    BV$BVA9,1          SPACE IN K3 IN-Q AND LBUS3 FREE?
       SAVEVALUE KXL3+,1            YES, RESERVE SLOT

************************
* USE LBUS3 SEND DATA TO  *
* K3                      *
************************

 UTX   MACRO     LBUS3,DYQ31,X$BEXD2

       SAVEVALUE DYL31-,1
       TRANSFER  ,RTF3              GO TO READ-THROUGH FROM L(3)

*********************
* READ FROM D32     *
*********************

 RRD32 QUEUE     DIQ32
       TEST E    BV$BVA22,1
       SAVEVALUE DYL32+,1

 UTX   MACRO     DRP32,DIQ32,X$DEX32

       QUEUE     DYQ32
       TEST E    BV$BVA9,1
       SAVEVALUE KXL3+,1

 UTX   MACRO     LBUS3,DYQ32,X$BEXD2

       SAVEVALUE DYL32-,1
       TRANSFER  ,RTF3

************************************
* READ-THROUGH FROM L(3) DATA IS   *
* SENT TO L(2) AND L(1) AT THE     *
* SAME TIME                        *
************************************

 RTF3  ASSIGN    11,0

************************
*  SERVICED BY K3      *
************************

 UQDQ  MACRO     KXQ3,KXL3-,KYQ3,KYL3+,KRP3,X$KEX,BV$BVA10

       TEST E    BV$RTOK,1          L(1) & L(2) READY & GBUS FREE?
       SAVEVALUE KXL1+,1
       SAVEVALUE KXL2+,1

************************
* BOTH L(1) AND L(2)   *
```

```
* READY TO ACCEPT DATA  *
* FROM GBUS             *
*************************

 UTX   MACRO     GBUS,KYQ3,X$BEXD2

       SAVEVALUE KYL3-,1
       SPLIT     1,STOR1          READ-THROUGH TO L(1)

*************************
* READ-THROUGH TO L(2)  *
*************************

 STOR2 ASSIGN    11,0

*************************
* SERVICED BY K2        *
*************************

 UQDQ  MACRO     KXQ2,KXL2-,KYQ2,KYL2+,KRP2,X$KEX,BV$BVA5

       TEST E    BV$BVA11,1       SPACE IN R2 IN-Q AND LBUS2 FREE?
       SAVEVALUE RXL2+,1          YES, RESERVE SLOT

*************************
* USE LBUS2 SEND TO R2  *
*************************

 UTX   MACRO     LBUS2,KYQ2,X$BEXD2

       SAVEVALUE KYL2-,1          FREE SLOT IN K2

*************************
* SERVICED BY R2        *
*************************

 UQD   MACRO     RXQ2,RXL2-,,RYL2+,RRP2,X$REX,BV$BVA12

       SPLIT     1,OVH2           HANDLE ANY OVERFLOW

*******************************
* STORE INTO D21 OR D22?      *
*******************************

       TRANSFER  .5,SSS21,SSS22

*************************
* STORE INTO D21        *
*************************

 SSS21 QUEUE     RYQ2
       TEST E    BV$BVA13,1       D21 IN-Q AND LBUS2 FREE?
       SAVEVALUE DXL21+,1         YES, RESERVE THE SPACE

*******************************
```

```
* SEND DATA TO D21 VIA BUS   *
******************************

 UTX   MACRO      LBUS2,RYQ2,X$BEXD2

       SAVEVALUE  RYL2-,1          RELEASE SPACE IN R2

 UQT   MACRO      DXQ21,DRP21,X$DEX21

       SAVEVALUE  DXL21-,1
       TERMINATE

*************************
* STORE INTO D22        *
*************************

 SSS22 QUEUE      RYQ2
       TEST E     BV$BVA23,1
       SAVEVALUE  DXL22+,1

 UTX   MACRO      LBUS2,RYQ2,X$BEXD2

       SAVEVALUE  RYL2-,1

 UQT   MACRO      DXQ22,DRP22,X$DEX22

       SAVEVALUE  DXL22-,1
       TERMINATE

*******************************
* HAND. ANY OVERF. FROM L(2) *
*******************************

 OVH2  TRANSFER   .X$POV2,NOV2,OVL2
 OVL2  QUEUE      ROQ2

*************************
* USE LBUS2, USE K2, USE  *
* GBUS, USE K3, USE LBUS3,  *
* THEN USE R3           *
*************************

 UTX   MACRO      LBUS2,ROQ2,X$BEXM

 UQTQ  MACRO      KIQ2,KRP2,KOQ2,X$KEX

 UTX   MACRO      GBUS,KOQ2,X$BEXM

 UQTQ  MACRO      KIQ3,KRP3,KOQ3,X$KEX

 UTX   MACRO      LBUS3,KOQ3,X$BEXM

 UQT   MACRO      RIQ3,RRP3,X$REX

 NOV2  TERMINATE
```

```
*************************
* COMMON CODE FOR WRITE *
* TO LOWER LEVELS       *
*************************
 COMW  ASSIGN    11,C              DUMMY STATEMENT

*************************
* SERVICED BY K1        *
*************************

 UQDQ  MACRO     KXQ1,KXL1-,KYQ1,KYL1+,KRP1,X$KEX,BV$BVA14

       TEST E    BV$BVA15,1        K2 IN-Q AND GBUS FREE?
       SAVEVALUE KXL2+,1

*************************
* USE GBUS              *
*************************

 UTX   MACRO     GBUS,KYQ1,X$BEXD1
       SAVEVALUE KYL1-,1

*************************
* SERVICED BY K2        *
*************************

 UQDQ  MACRO     KXQ2,KXL2-,KYQ2,KYL2+,KRP2,X$KEX,BV$BVA5

       TEST E    BV$BVA11,1        R2 IN-Q AND LBUS2 FREE?
       SAVEVALUE RXL2+,1

*************************
* USE LBUS2             *
*************************

 UTX   MACRO     LBUS2,KYQ2,X$BEXD1
       SAVEVALUE KYL2-,1

*************************
* SERVICED BY R2        *
*************************

 UQD   MACRO     RXQ2,RXL2-,,RYL2+,RRP2,X$REX,BV$BVA12

*********************************
* SERVED BY D21 OR D22?         *
*********************************

       TRANSFER  .5,SWS21,SWS22


*************************
* SERVICED BY D21       *
*************************
...
```

```
SWS21 QUEUE     RYQ2
      TEST E    BV$BVA13,1
      SAVEVALUE DXL21+,1

UTX   MACRO     LBUS2,RYQ2,X$BEXD1

      SAVEVALUE RYL2-,1
UQDQ  MACRO     DXQ21,DXL21-,DYQ21,DYL21+,DRP21,X$DEX21,BV$BVA3

      TEST E    BV$BVA4,1         K2 IN-Q AND LBUS2 FREE?
      SAVEVALUE KXL2+,1

*************************
* USE LBUS2 SEND TO K2  *
*************************

UTX   MACRO     LBUS2,DYQ21,X$BEXD2

      SAVEVALUE DYL21-,1
      SPLIT     1,ACK2            PREPARE TO SEND ACK TO L(1)

      TRANSFER  ,STD23            GO TO STORE-BEHIND TO L(3)

*******************************
* SEND ACK TO L(1)            *
*******************************

ACK2  QUEUE     DOQ21

UTX   MACRO     LBUS2,DOQ21,X$BEXN

      TRANSFER  ,ACK21

*******************************
* SERVICED BY D22             *
*******************************

SWS22 QUEUE     RYQ2
      TEST E    BV$BVA23,1
      SAVEVALUE DXL22+,1

UTX   MACRO     LBUS2,RYQ2,X$BEXD1

      SAVEVALUE RYL2-,1

UQDQ  MACRO     DXQ22,DXL22-,DYQ22,DYL22+,DRP22,X$DEX22,BV$BVA21

      TEST E    BV$BVA4,1
      SAVEVALUE KXL2+,1

UTX   MACRO     LBUS2,DYQ22,X$BEXD2

      SAVEVALUE DYL22-,1
      SPLIT     1,ACK3
```

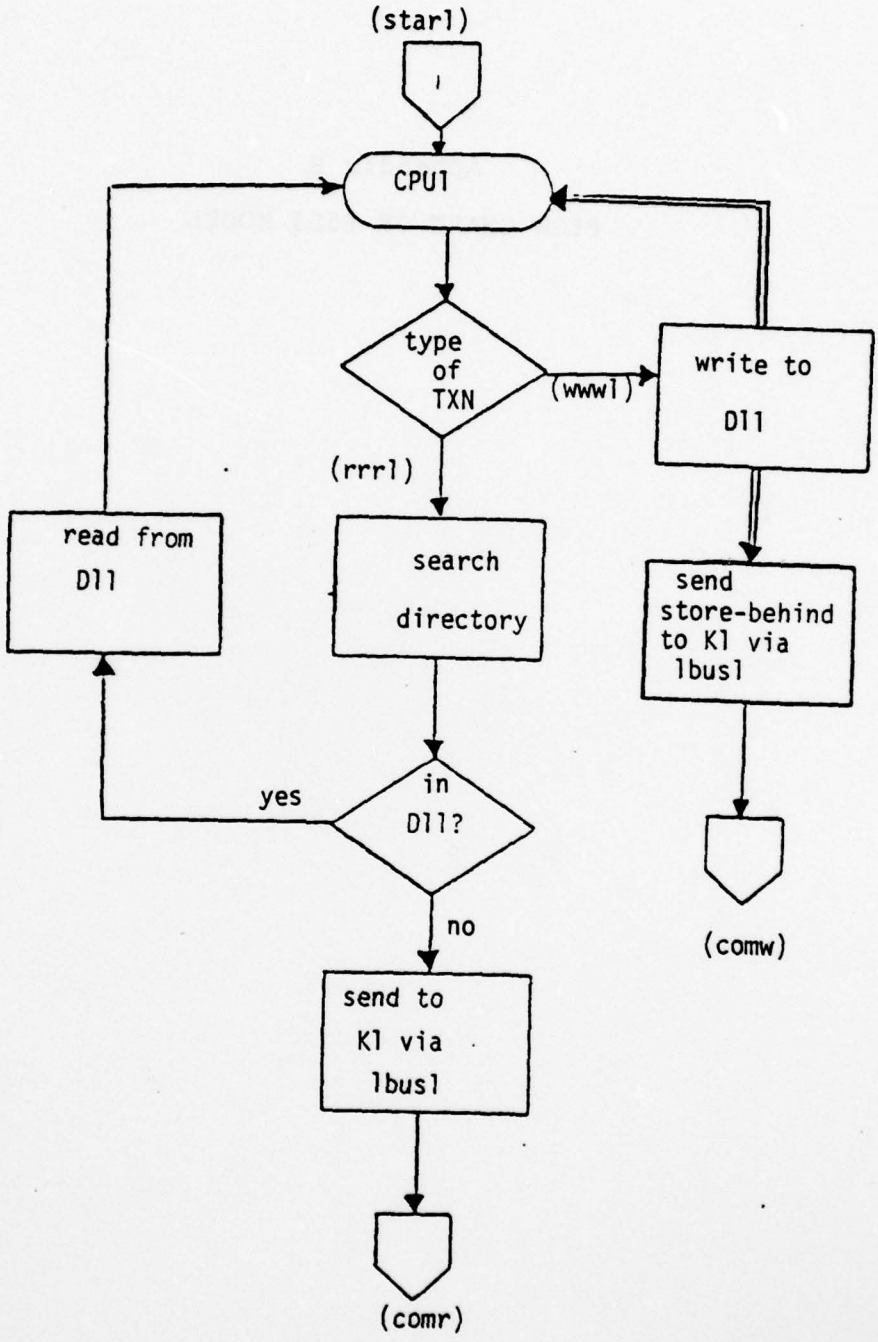
```
         TRANSFER   ,STB23

ACK3     QUEUE      DOQ22

UTX      MACRO      LBUS2,DOQ22,X$BEXM

         TRANSFER   ,ACK21


*************************
* STORE-BEHIND FROM     *
* L(2) TO L(3)          *
*************************

STB23    ASSIGN     11,0

UQDQ     MACRO      KXQ2,KXL2-,KYQ2,KYL2+,KRP2,X$KEX,BV$BVA5

         TEST E     BV$BVA16,1        K3 IN-Q AND GBUS FREE?
         SAVEVALUE  KXL3+,1

UTX      MACRO      GBUS,KYQ2,X$BEXD2

         SAVEVALUE  KYL2-,1

UQDQ     MACRO      KXQ3,KXL3-,KYQ3,KYL3+,KRP3,X$KEX,BV$BVA10

         TEST E     BV$BVA17,1        R3 IN-Q AND LBUS3 FREE?
         SAVEVALUE  RXL3+,1

UTX      MACRO      LBUS3,KYQ3,X$BEXD2

         SAVEVALUE  KYL3-,1

UQD      MACRO      RXQ3,RXL3-,,RYL3+,RRP3,X$REX,BV$BVA18

************************************
* SERVICED BY D31 OR D32?          *
************************************

         TRANSFER   .5,SWS31,SWS32

*********************
* SERV. BY D31      *
*********************

SWS31    QUEUE      RYQ3
         TEST E     BV$BVA19,1
         SAVEVALUE  DXL31+,1

UTX      MACRO      LBUS3,RYQ3,X$BEXD2

         SAVEVALUE  RYL3-,1
```

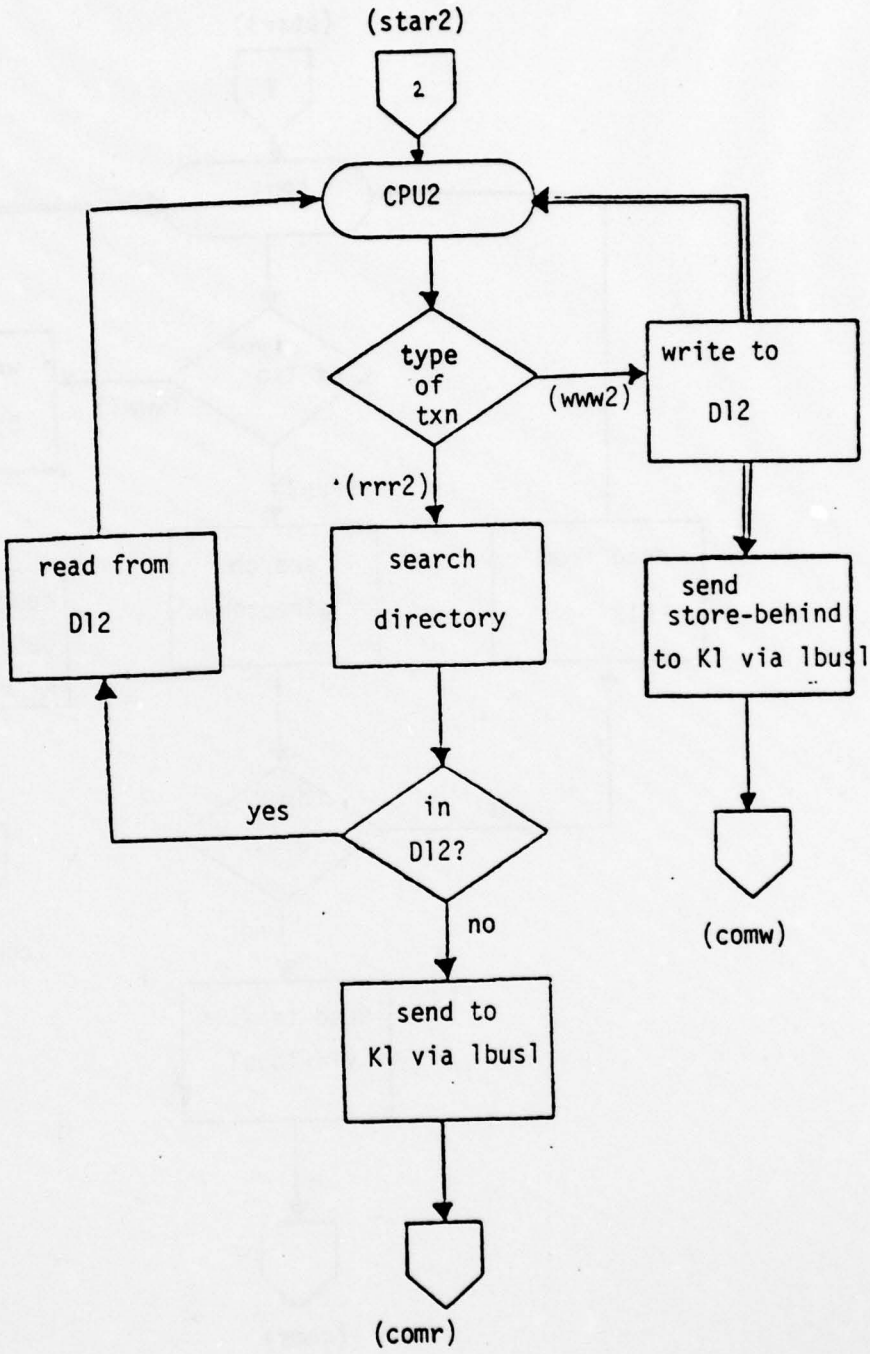
```
 UQT    MACRO       DXQ31,DRP31,X$DEX31
        SAVEVALUE   DXL31-,1
 UQT    MACRO       DOQ31,LBUS3,X$BEXM
        TRANSFER    ,ACK22
**********************
* SERV. BY D32  *
**********************
 SWS32  QUEUE       RYQ3
        TEST E      BVSBVA24,1
        SAVEVALUE   DXL32+,1
 UTX    MACRO       LBUS3,RYQ3,X$BEXD2
        SAVEVALUE   RYL3-,1
 UQT    MACRO       DXQ32,DRP32,X$DEX32
        SAVEVALUE   DXL32-,1
 UQT    MACRO       DOQ32,LBUS3,X$BEXM
        TRANSFER    ,ACK22

************************************
* ACK FROM L(2) TO L(3)            *
************************************
 ACK22  ASSIGN      11,0
 UQTQ   MACRO       KIQ3,KRP3,KOQ3,X$KEX
 UTX    MACRO       GBUS,KOQ3,X$BEXM
 UQTQ   MACRO       KIQ2,KRP2,KOQ2,X$KEX
 UTX    MACRO       LBUS2,KOQ2,X$BEXM
 UQTQ   MACRO       RIQ2,RRP2,ROQ2,X$REX
 UTX    MACRO       LBUS2,ROQ2,X$BEXM
        TRANSFER    ,ACK21
************************************
* ACK FROM L(2) TO L(1)            *
************************************
 ACK21  ASSIGN      11,0
 UQTQ   MACRO       KIQ2,KRP2,KOQ2,X$KEX
```

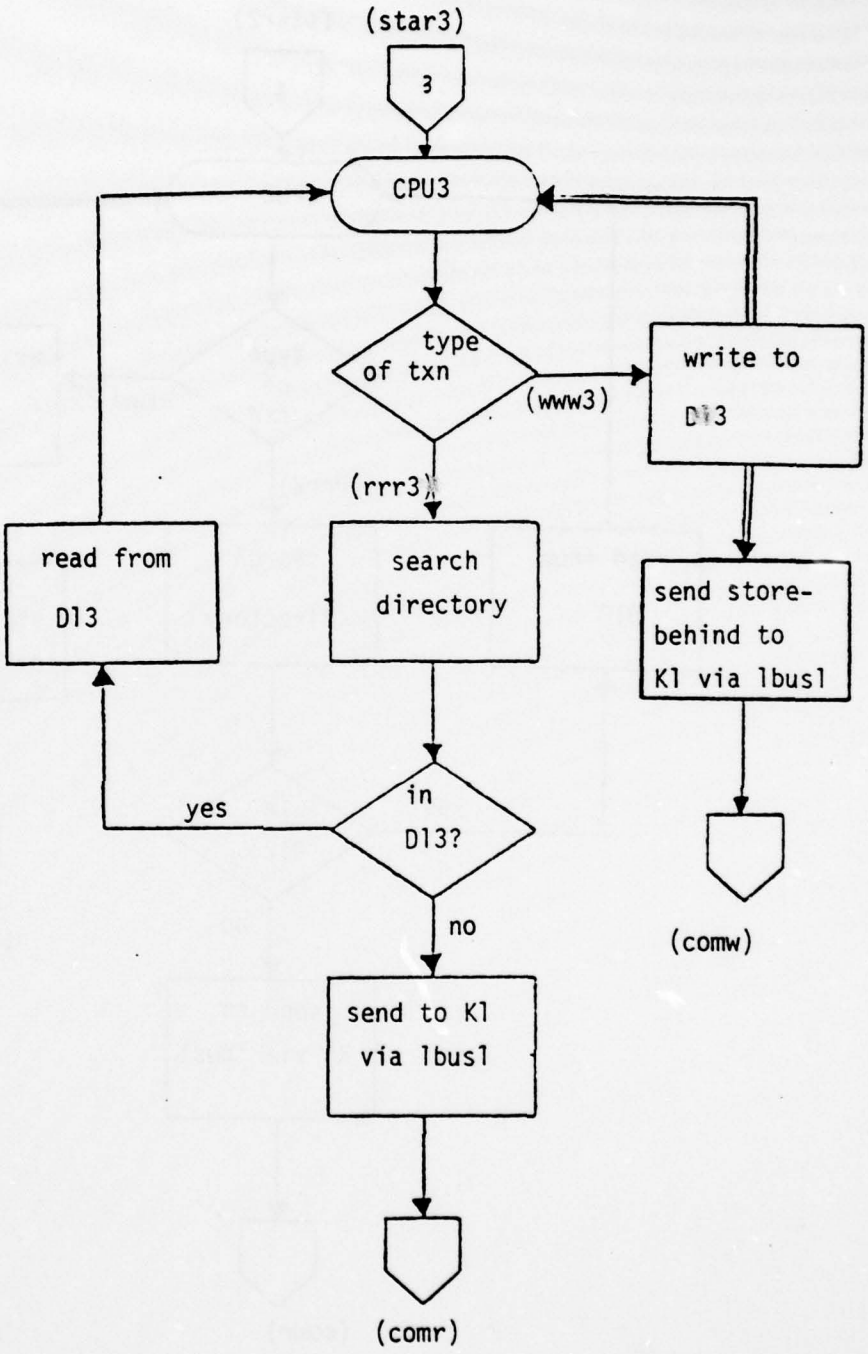
```
 UTX    MACRO      GBUS,KOQ2,XSBEXM

 UQTQ   MACRO      KIQ1,KRP1,KOQ1,XSKEX

 UTX    MACRO      LBUS1,KOQ1,XSDEXM
        SPLIT      1,FNSWICHA,1
        TERMINATE
 AAA11  ASSIGN     11,0
 AAA12  ASSIGN     11,0
 AAA13  ASSIGN     11,0
        QUEUE      DIQ11
        SEIZE      DRP11
        DEPART     DIQ11
        ASSIGN     4+,XSREX
        ASSIGN     7,XSREX
        ADVANCE    P7
        RELEASE    DRP11
        TERMINATE

******************
* AAA12          *
******************

******************
*  AAA13         *
******************


************************************
*                                  *
*  TIMER SEGMENT - TIME UNIT  IS   *
*  ONE NANOSECOND                  *
************************************

        GENERATE   XSTIMER
        TERMINATE  1

        START      1
        END
/*
```

Appendix B

FLOW CHART OF P5L3 MODEL


(star1)
CPU1
type of TXN
(www1)
write to D11
(rrr1)
read from D11
search directory
send store-behind to K1 via lbus1
yes
in D11?
no
(comw)
send to K1 via lbus1
(comr)

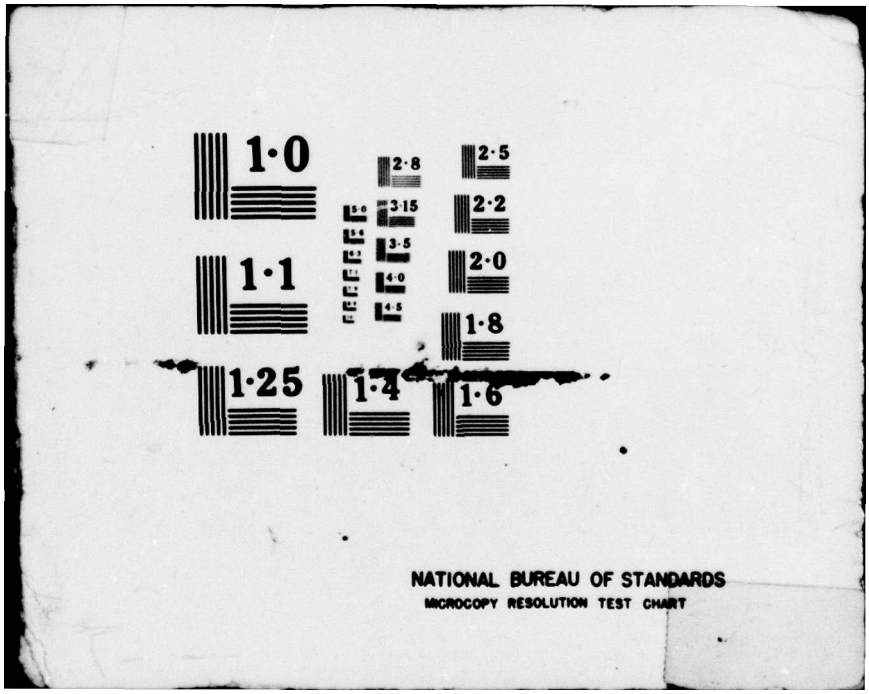

(star2)
2
CPU2
type of txn
(www2)
write to D12
(rrr2)
read from D12
search directory
send store-behind to K1 via lbus1
yes
in D12?
(comw)
no
send to K1 via lbus1
(comr)

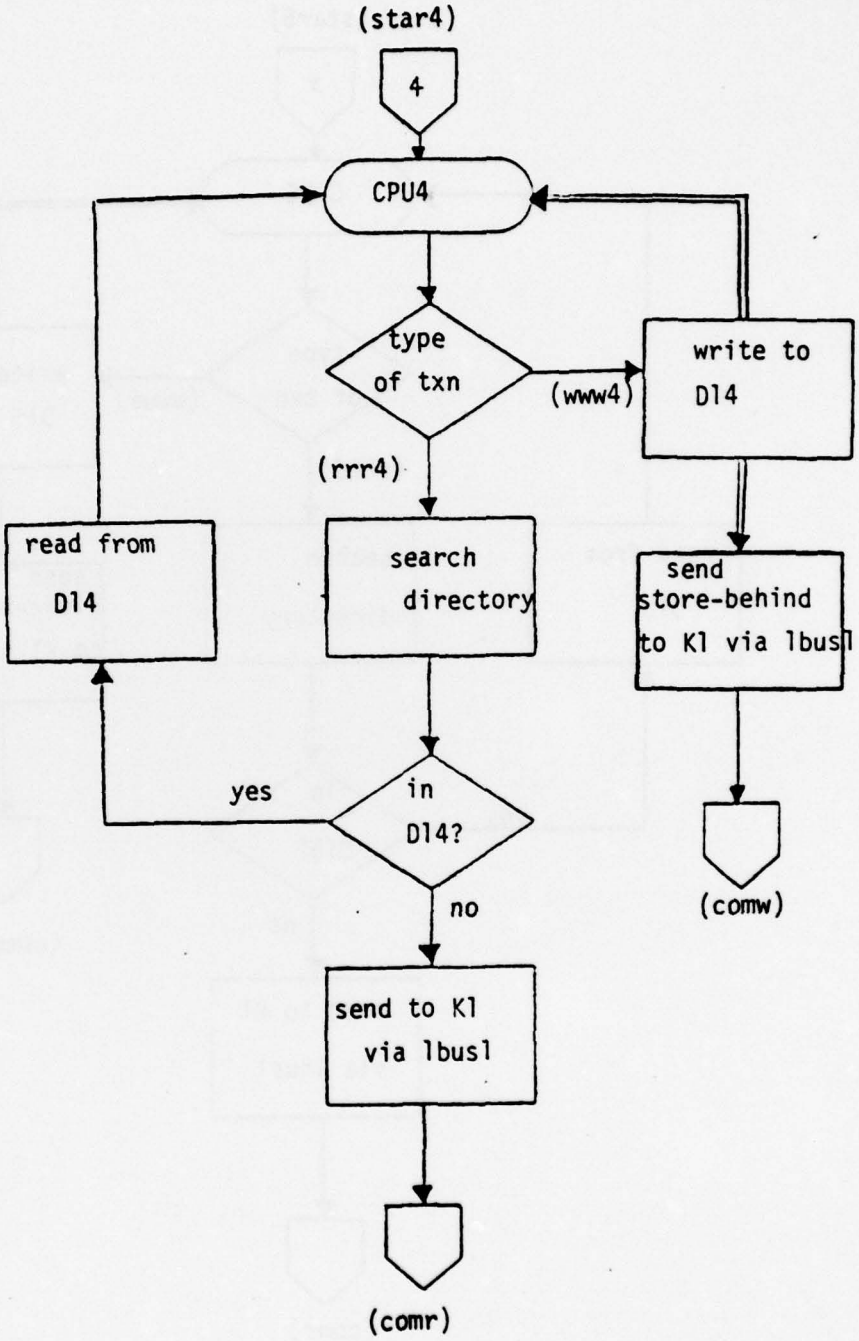

(star3)
3
CPU3
type of txn
(www3)
write to D13
(rrr3)
read from D13
search directory
send store-behind to K1 via lbus1
yes
in D13?
no
(comw)
send to K1 via lbus1
(comr)

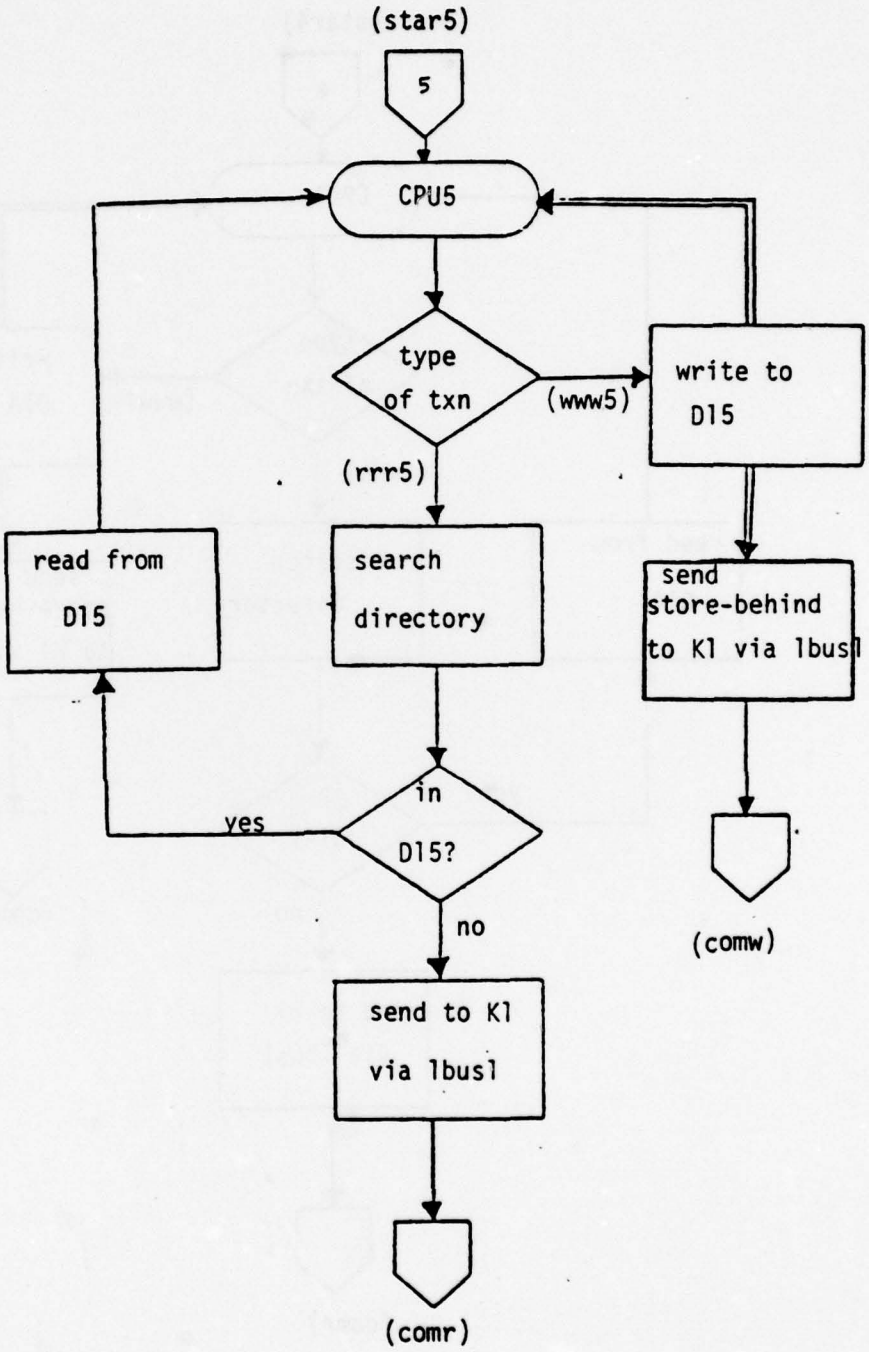
AD-A074 503 ALFRED P SLOAN SCHOOL OF MANAGEMENT CAMBRIDGE MA CEN--ETC F/G 9/2
SIMULATION STUDIES OF THE DSH-11 DATA STORAGE HIERARCHY SYSTEM.(U)
SEP 79 C LAM, S E MADNICK N00039-78-G-0160
UNCLASSIFIED CISR-M010-7909-04 NL

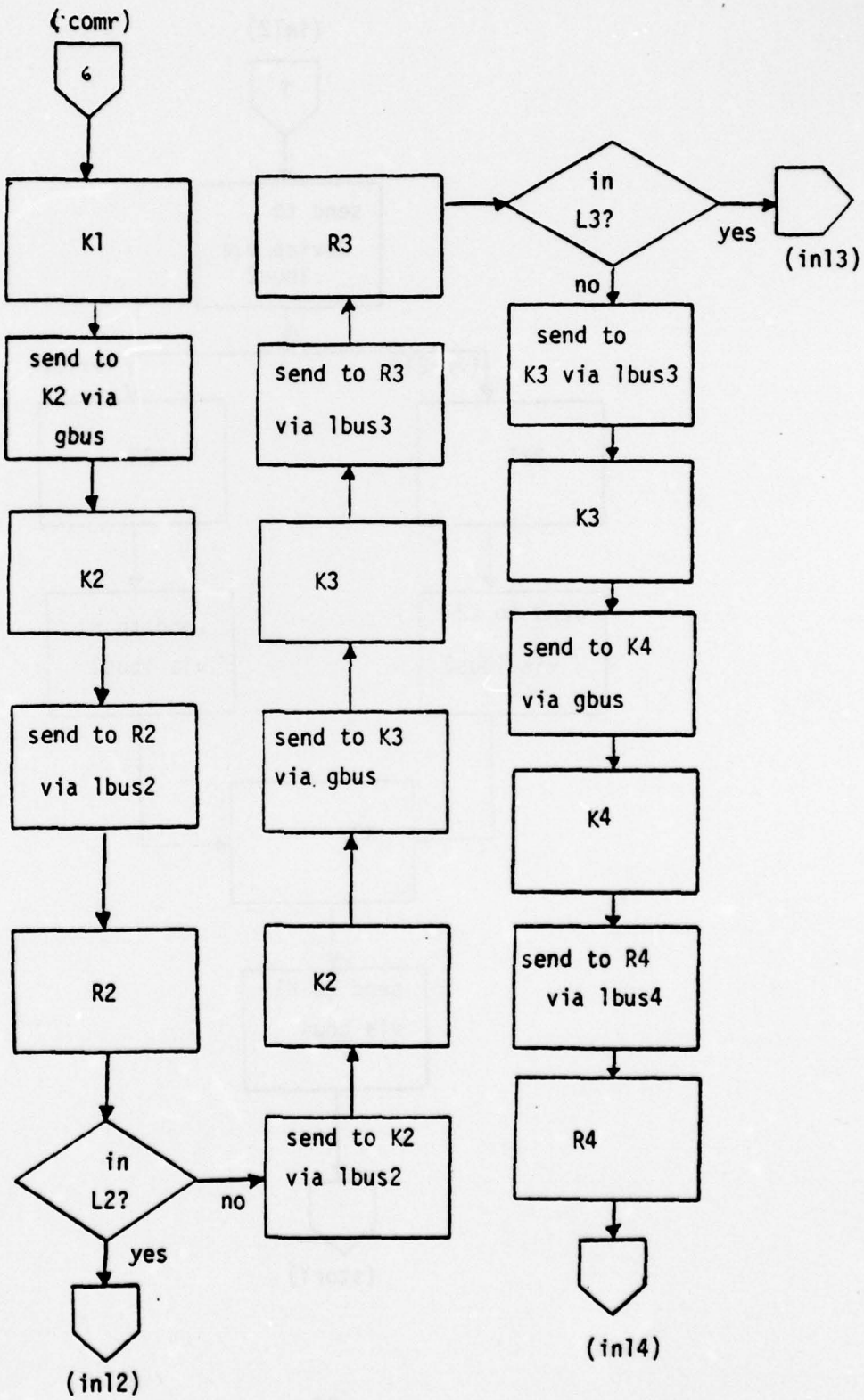
2 OF 2
AD-
A074503

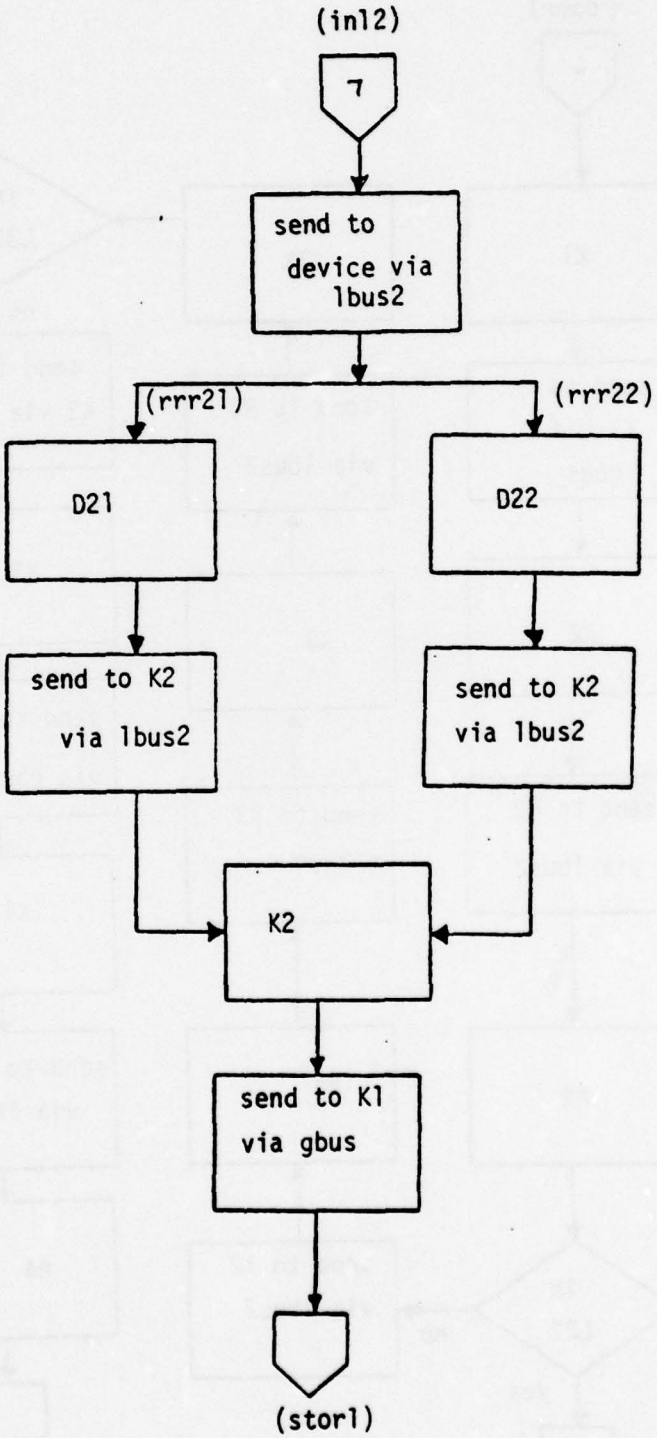
END
DATE FILMED
11-79
DDC

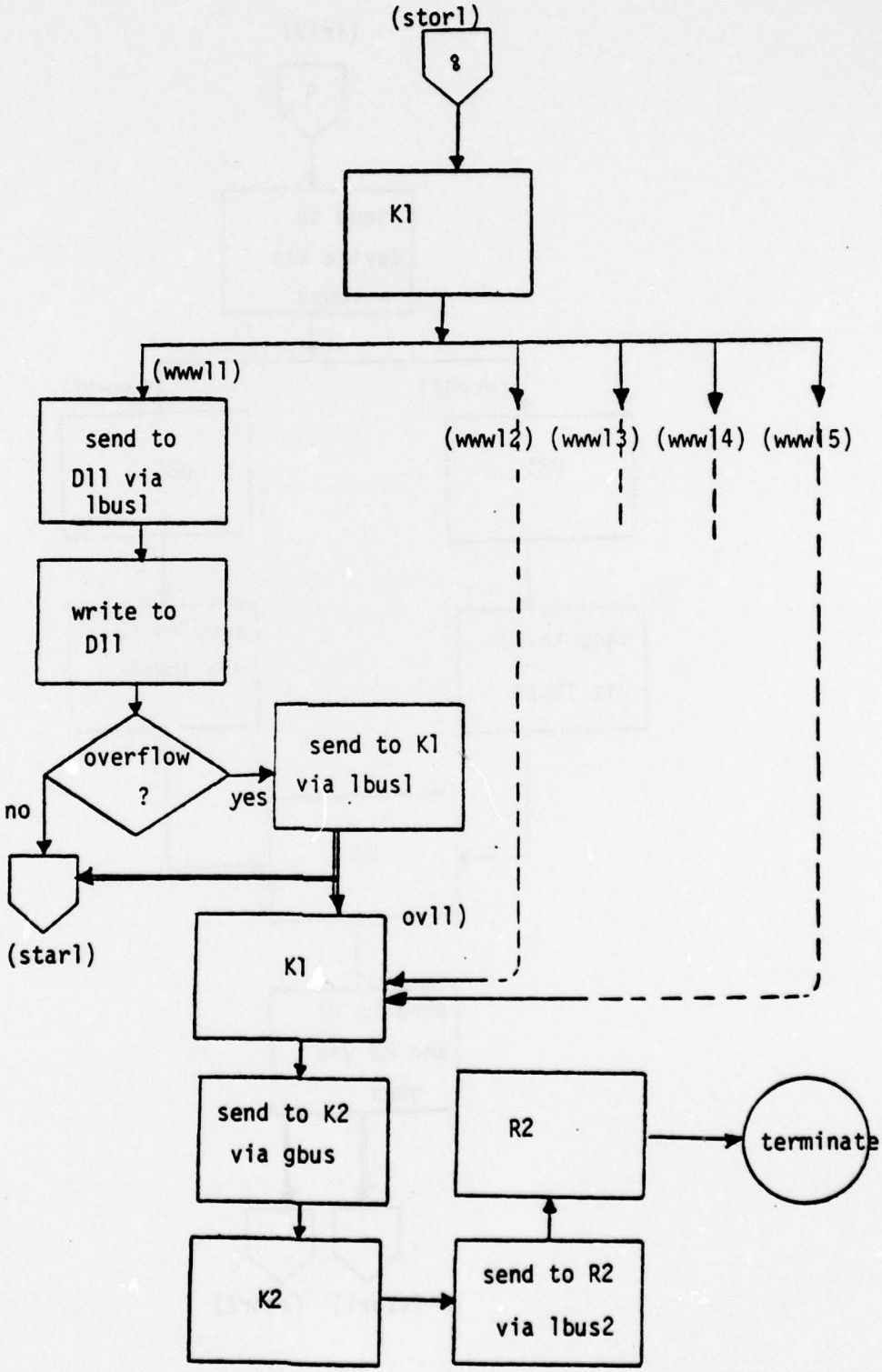

1·0
1·1
1·25
2·8
3·15
3·5
4·0
4·5
1·4
2·5
2·2
2·0
1·8
1·6
NATIONAL BUREAU OF STANDARDS
MICROCOPY RESOLUTION TEST CHART

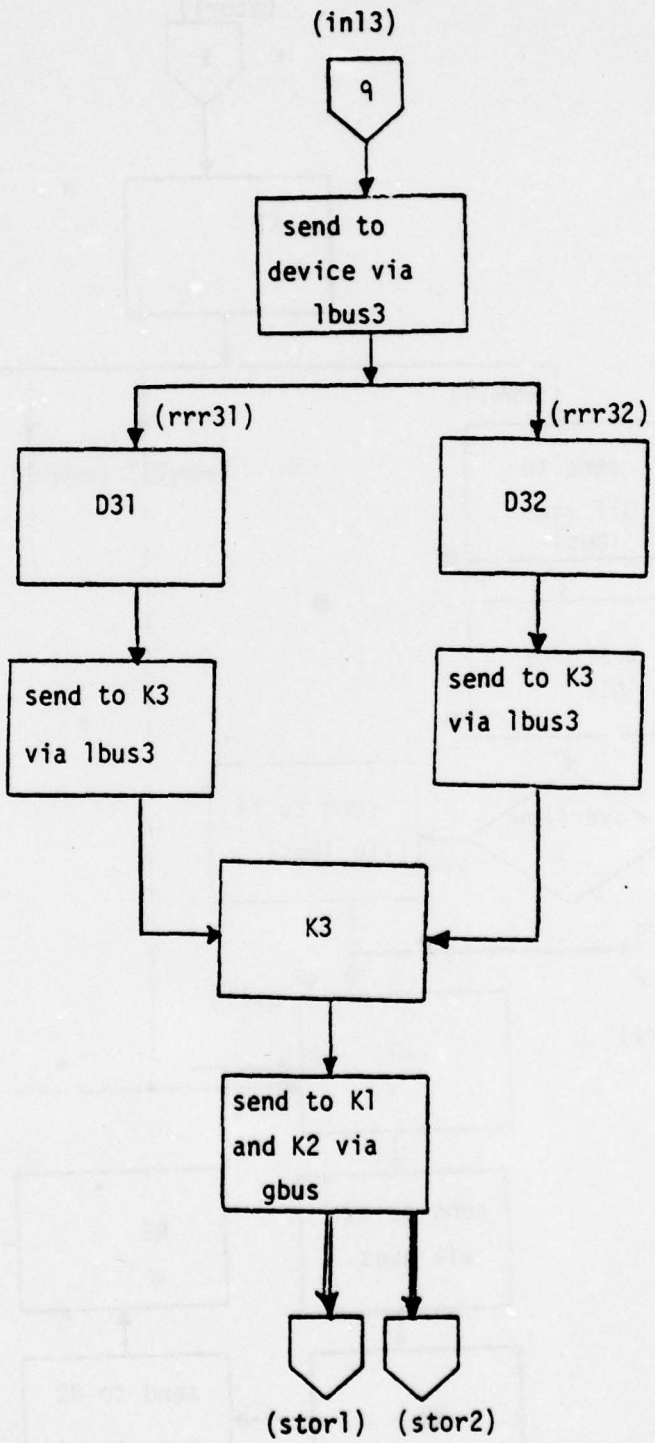

(star4)
4
CPU4
type
of txn
(www4)
write to
D14
(rrr4)
read from
D14
search
directory
send
store-behind
to K1 via lbus1
yes
in
D14?
no
(comw)
send to K1
via lbus1
(comr)

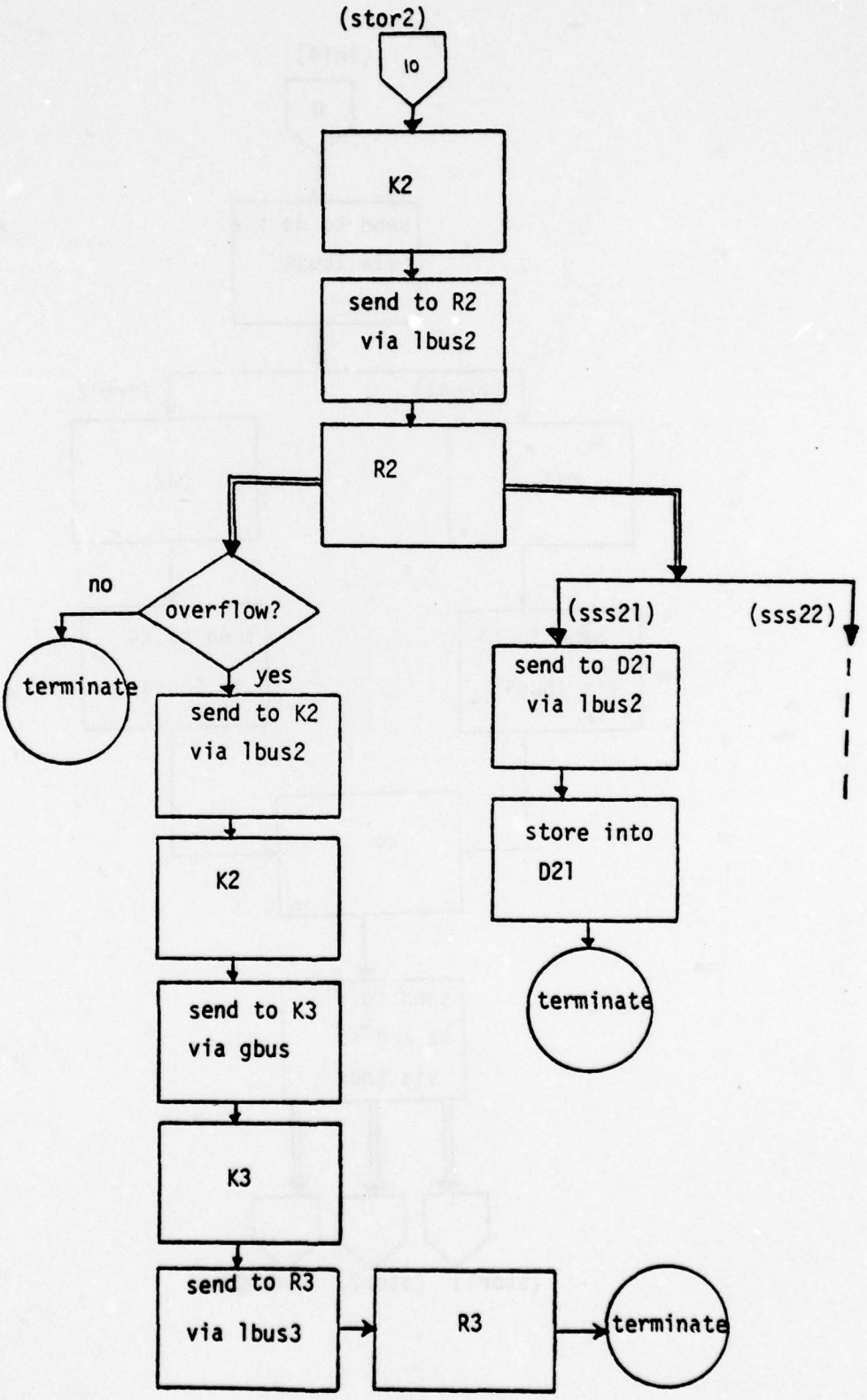

(star5)
5
CPU5
type
of txn
(www5)
write to
D15
(rrr5)
read from
D15
search
directory
send
store-behind
to K1 via lbus1
in
D15?
yes
no
(comw)
send to K1
via lbus1
(comr)

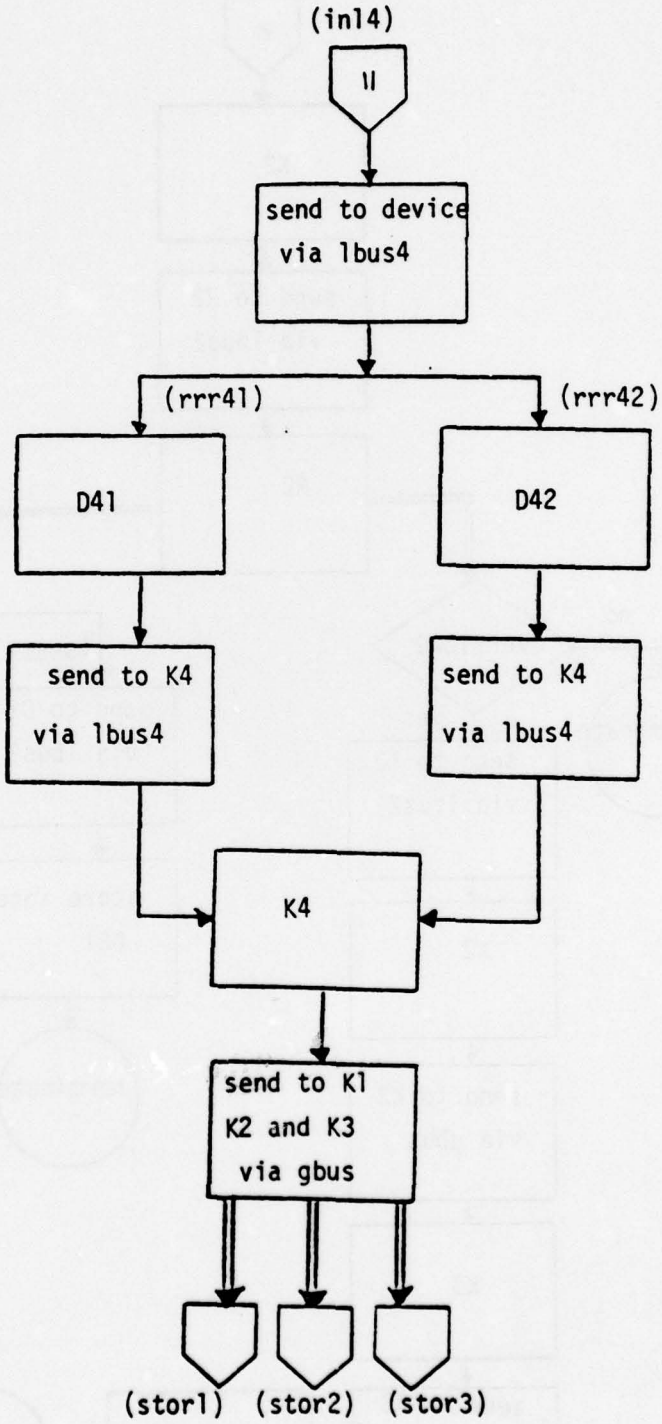

('comr)
6
K1
send to K2 via gbus
K2
send to R2 via lbus2
R2
in L2?
yes
(in12)
no
send to K2 via lbus2
K2
send to K3 via gbus
K3
send to R3 via lbus3
R3
in L3?
yes
(in13)
no
send to K3 via lbus3
K3
send to K4 via gbus
K4
send to R4 via lbus4
R4
(in14)

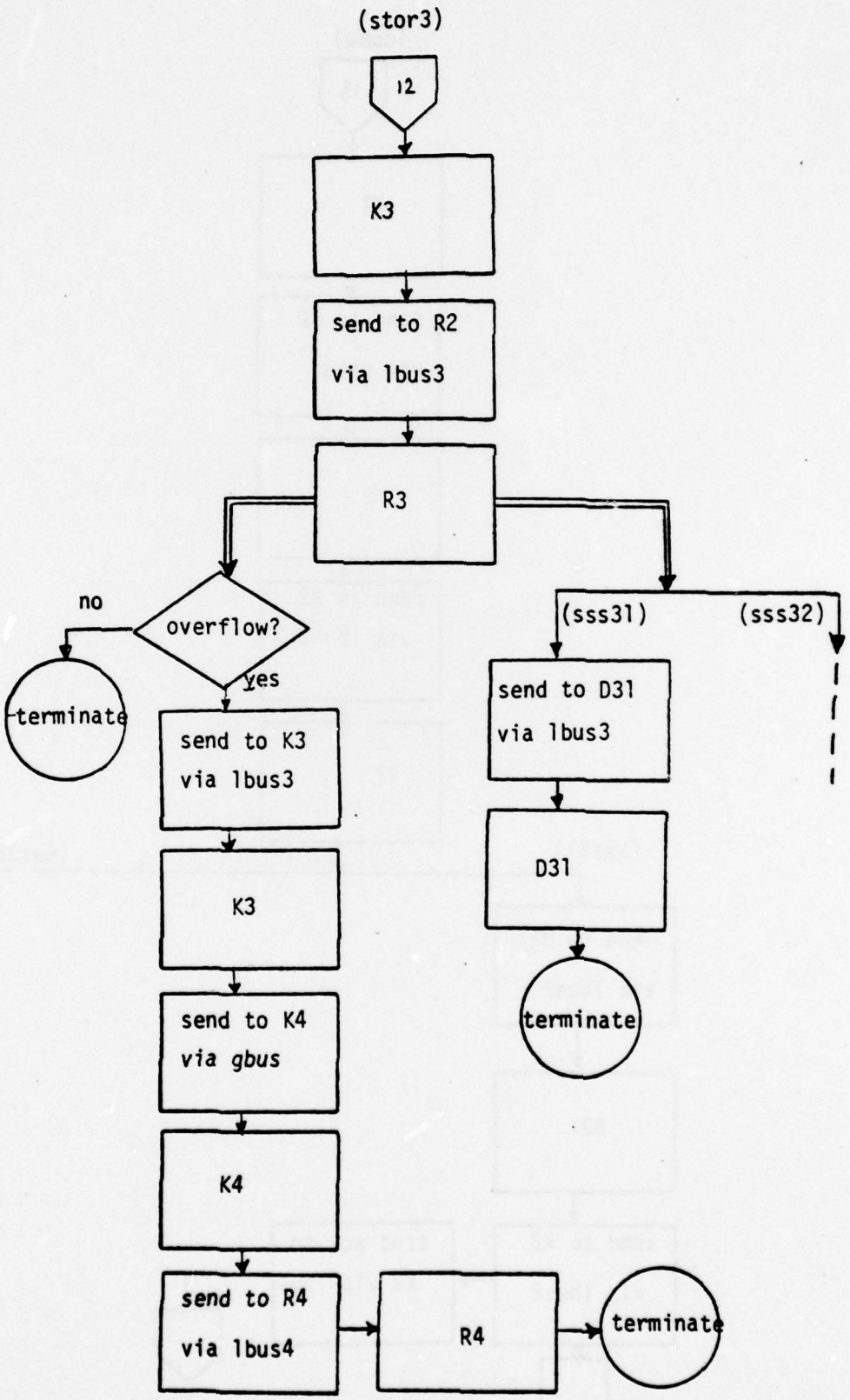

(in12)
7
send to
device via
1bus2
(rrr21)
(rrr22)
D21
D22
send to K2
via 1bus2
send to K2
via 1bus2
K2
send to K1
via gbus
(stor1)

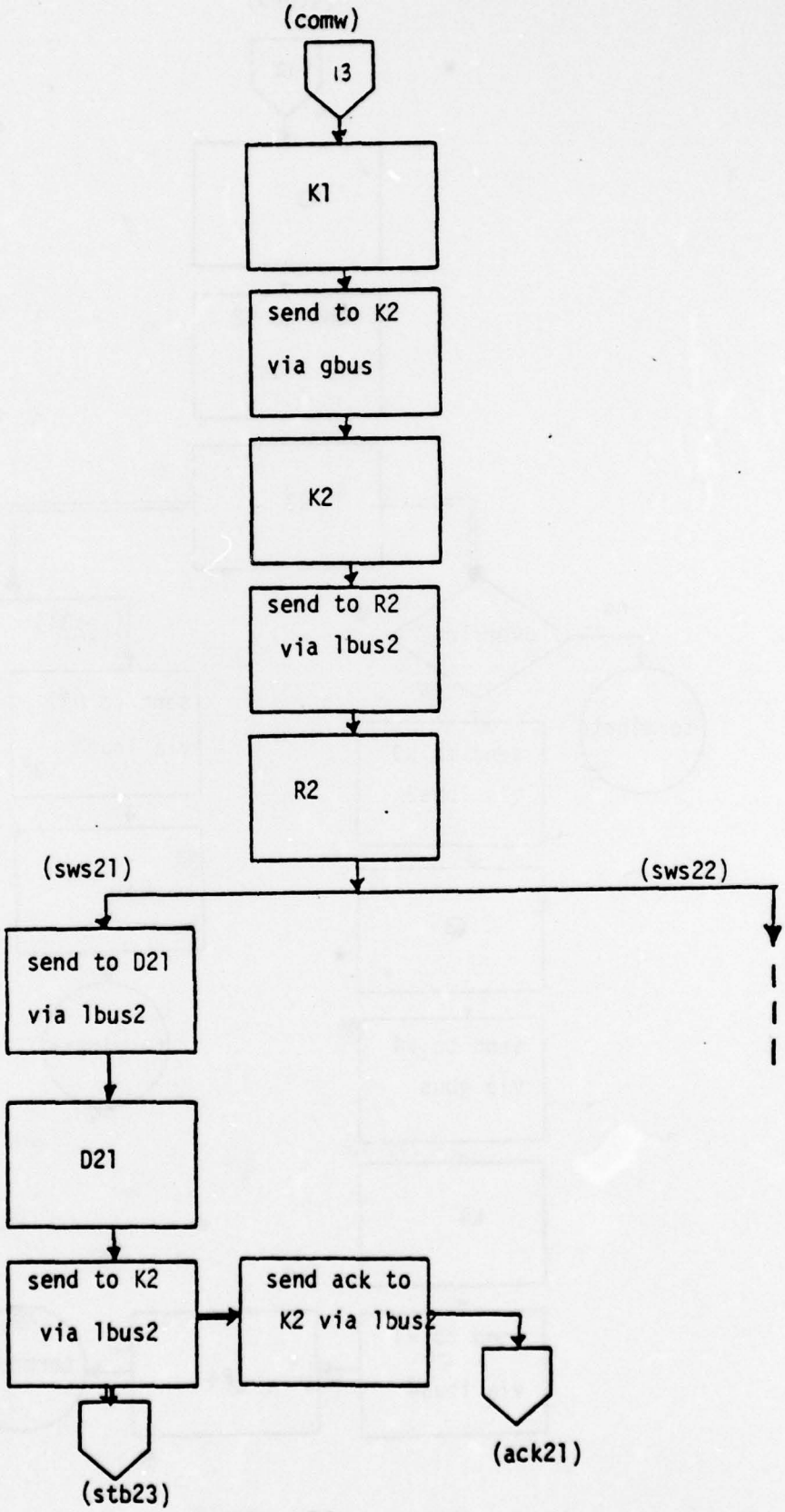

(stor1)
8
K1
(www11)
send to
D11 via
lbus1
write to
D11
overflow
?
no
yes
send to K1
via lbus1
(star1)
ov11)
K1
(www12)
(www13)
(www14)
(www15)
send to K2
via gbus
K2
R2
send to R2
via lbus2
terminate

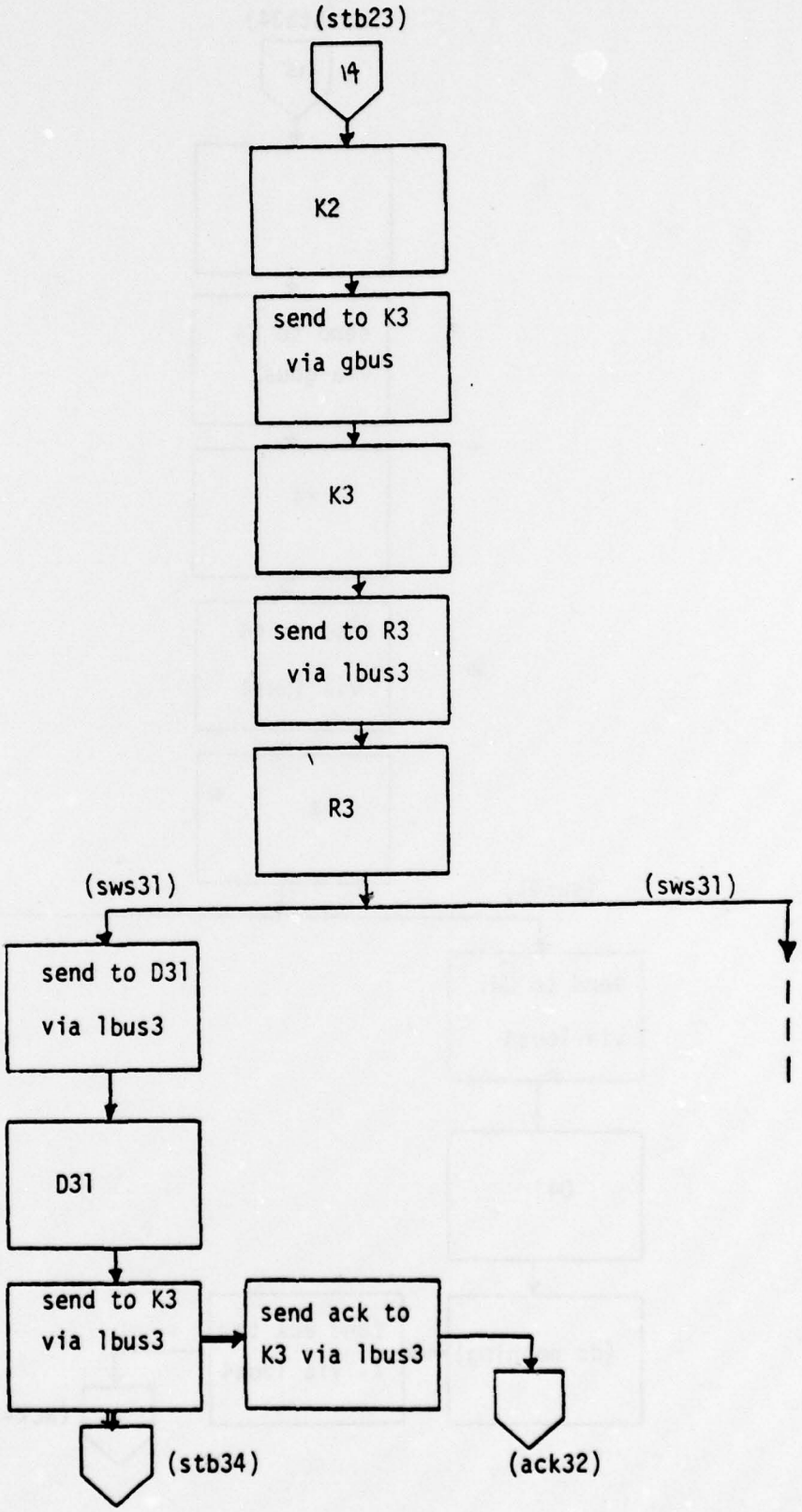

(in13)
9
send to device via 1bus3
(rrr31)
(rrr32)
D31
D32
send to K3 via 1bus3
send to K3 via 1bus3
K3
send to K1 and K2 via gbus
(stor1)
(stor2)

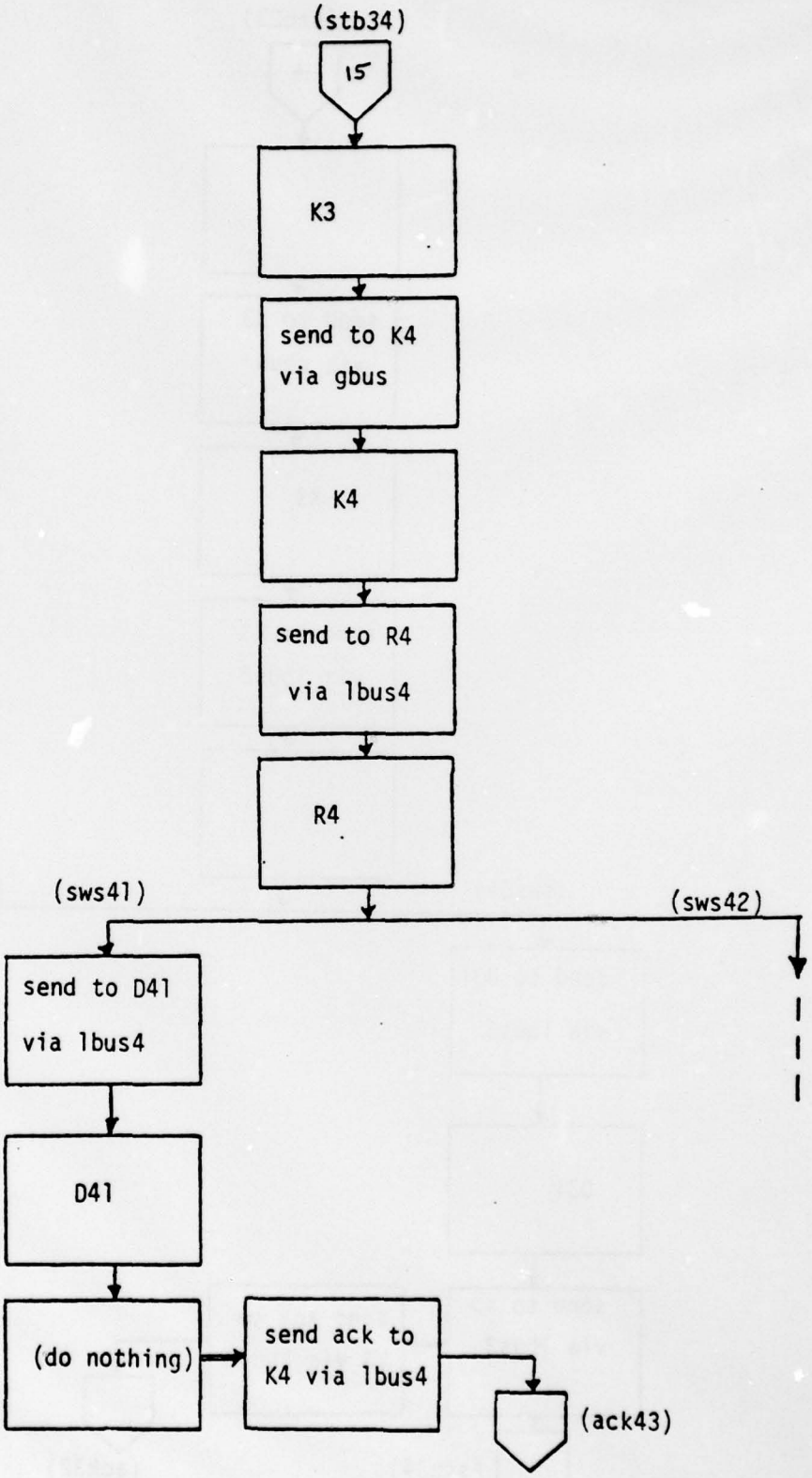

(stor2)
10
K2
send to R2
via lbus2
R2
no
overflow?
yes
terminate
send to K2
via lbus2
K2
send to K3
via gbus
K3
send to R3
via lbus3
R3
terminate
(sss21)
(sss22)
send to D21
via lbus2
store into
D21
terminate

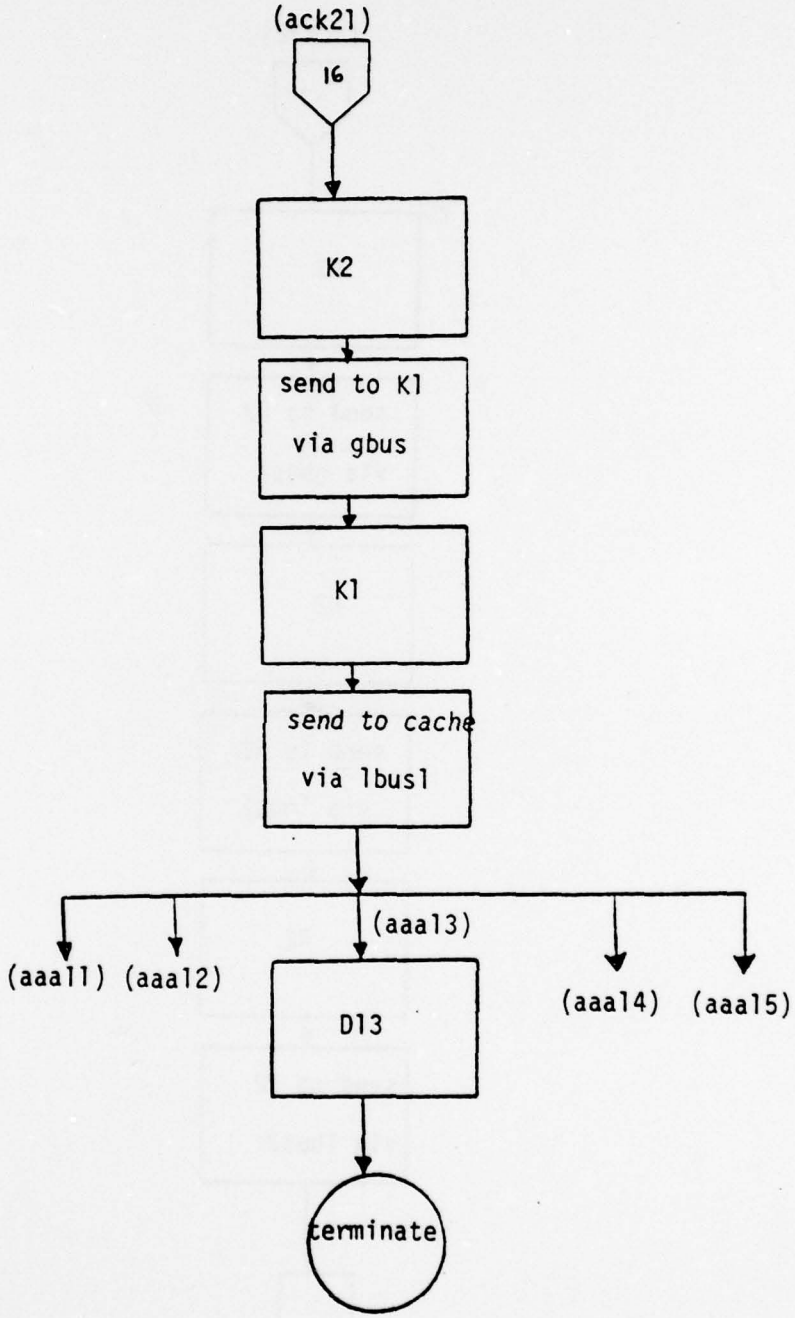

(in14)
11
send to device
via lbus4
(rrr41)
(rrr42)
D41
D42
send to K4
via lbus4
send to K4
via lbus4
K4
send to K1
K2 and K3
via gbus
(stor1)
(stor2)
(stor3)

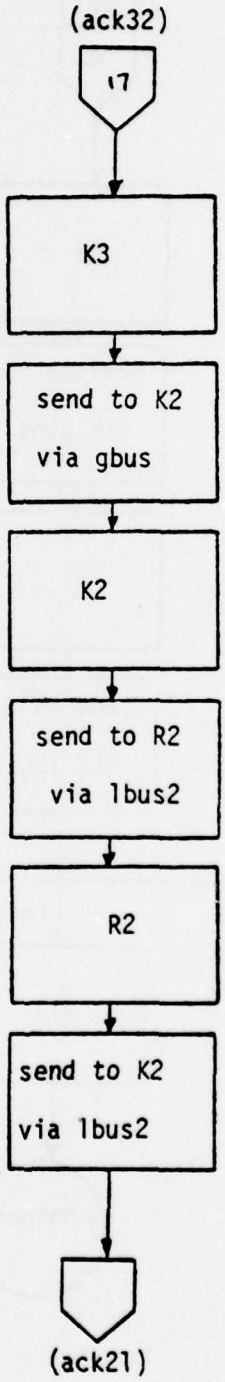

(stor3)
12
K3
send to R2
via lbus3
R3
no
overflow?
yes
terminate
send to K3
via lbus3
K3
send to K4
via gbus
K4
send to R4
via lbus4
R4
terminate
(sss31)
(sss32)
send to D31
via lbus3
D31
terminate

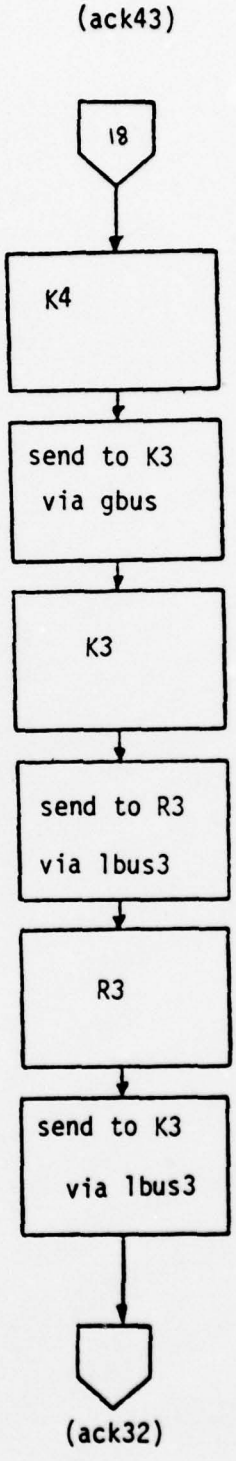

(comw)
13
K1
send to K2
via gbus
K2
send to R2
via lbus2
R2
(sws21)
(sws22)
send to D21
via lbus2
D21
send to K2
via lbus2
send ack to
K2 via lbus2
(ack21)
(stb23)


(stb23)
14
K2
send to K3
via gbus
K3
send to R3
via lbus3
R3
(sws31)
(sws31)
send to D31
via lbus3
D31
send to K3
via lbus3
send ack to
K3 via lbus3
(stb34)
(ack32)


(stb34)
15
K3
send to K4
via gbus
K4
send to R4
via lbus4
R4
(sws41)
(sws42)
send to D41
via lbus4
D41
(do nothing)
send ack to
K4 via lbus4
(ack43)


(ack21)
16
K2
send to K1
via gbus
K1
send to cache
via lbus1
(aaa11)
(aaa12)
(aaa13)
(aaa14)
(aaa15)
D13
terminate


(ack32)
17
K3
send to K2
via gbus
K2
send to R2
via lbus2
R2
send to K2
via lbus2
(ack21)


(ack43)
18
K4
send to K3
via gbus
K3
send to R3
via lbus3
R3
send to K3
via lbus3
(ack32)

Appendix C

LISTING OF THE P5L4 MODEL

FILE: GPSS54 VS1JOB D1 CONVERSATIONAL MONITOR SYSTEM

```
//LAM4 JOB LAM,MSGPROFILE='RETURN',
// PROFILE='LOW',
// TIME=9
//*PASSWORD
//GPSS PROC
//C   EXEC  PGM=DAG01,TIME=&TLIMIT
//STEPLIB DD DSN=POTLUCK.LIBRARY.GPSS.LOAD,DISP=SHR
//DOUTPUT DD SYSOUT=PROFILE=RETURN,DCB=BLKSIZE=931
//DINTERO DD UNIT=SCRATCH,SPACE=(CYL,(1,1)),DCB=BLKSIZE=1880
//DSYMTAB DD UNIT=SCRATCH,SPACE=(CYL,(1,1)),DCB=BLKSIZE=7112
//DRCPTGEN DD UNIT=SCRATCH,SPACE=(CYL,(1,1)),DCB=BLKSIZE=800
//DINTWORK DD UNIT=SCRATCH,SPACE=(CYL,(1,1)),DCB=BLKSIZE=2680
// PEND
//STEP1  EXEC  GPSS,PARM=C,TLIMIT=9
//DINPUT1  DD *
       REALLOCATE FUN,5,QUE,10,FAC,50,BVR,200,BLO,2000,VAR,50
       REALLOCATE FSV,50,HSV,10,COM,40000

**************************
*                        *
* TXN PARM USAGE         *
*                        *
*  P1   CPU ID           *
*  P2   TXN ARRIVAL TIME *
*  P3   TXN COMPL TIME   *
*  P4   TXN EXEC TIME    *
*  P11  DUMMY            *
*                        *
**************************

**************************
*                        *
*  MODEL COMPONENTS      *
*                        *
* BUSES: GBUS, LBUS1,..  *
* CACHES: D11,...D15     *
* LEVEL CONTRL: K1,...K4 *
* REQ PROCS: R2, .. R4   *
* DEVICES: D21, ...D42   *
* STORAGE : RI, RO       *
* STORAGE : SI, SO       *
* STORAGE : TI, TO       *
* STORAGE : AI, AO       *
* STORAGE : OI, OO       *
*                        *
**************************

**************************
*                        *
* MODEL PARAMETERS       *
*                        *
**************************

       INITIAL    XSMAXMP,10        DEGREE OF MULTIPROG PER CPU
       INITIAL    XSNREAD,500       % READ REQ
```

```
         INITIAL    X$NWRIT,500       % WRITE REQ
         INITIAL    X$PIN1,900        CONDITIONAL PROB OF FINDING DATA
         INITIAL    X$PIN2,900        IN A LEVEL GIVEN THAT THE
         INITIAL    X$PIN3,900        DATA IS NOT FOUND IN ANY UPPER
         INITIAL    X$PIN4,1000       LEVEL
         INITIAL    X$POV1,500        PROB OF OVERFLOW
         INITIAL    X$POV2,500
         INITIAL    X$POV3,500
         INITIAL    X$DEX1,10       DEVICE SERVICE TIME
         INITIAL    X$DEX2,100
         INITIAL    X$DEX3,200
         INITIAL    X$DEX4,1000
         INITIAL    X$BEXM,10       BUS SERVICE TIME
         INITIAL    X$BEX1,10
         INITIAL    X$BEX2,80
         INITIAL    X$BEX3,320
         INITIAL    X$DEX,20        DIRECTORY LOOK UP
         INITIAL    X$KEX,10        CONTROLLER SERV TIME
         INITIAL    X$RDEX1,30      LOOKUP PLUS READ TIME OF CACHE
         INITIAL    X$TIMER,200000  SIMULATION TIME

**************************
*                        *
* SAVEVALUES             *
*                        *
* NTXN TOTAL TXN PROC.   *
* SUMX TOTAL EXEC TIMES  *
* SUMW TOTAL WAIT TIMES  *
* SUMT TOTAL ELAPSED TIM*
*                        *
**************************

**************************
*                        *
*  VARIABLES             *
*                        *
**************************

 MRESP FVARIABLE  (X$SUMT/X$NTXN)      MEAN RESP TIME
 TXNT  VARIABLE   P3-P2                TXN ELAPSED TIME
 TXNW  VARIABLE   P3-P2-P4             TXN WAIT TIME
 TXNX  VARIABLE   P4                   TXN EXEC TIME

**************************
*                        *
*  TABLES                *
*                        *
**************************

 TXNT  TABLE      VSTXNT,100,100,100
 TXNW  TABLE      VSTXNW,100,100,100
 TXNX  TABLE      VSTXNX,100,100,100

**************************
*                        *
```

```
*  FUNCTIONS            *
*                       *
*************************

 WICHW FUNCTION   P1,D5
2,WWW11/3,WWW12/4,WWW13/5,WWW14/6,WWW15

 WICHA FUNCTION   P1,D5
2,AAA11/3,AAA12/4,AAA13/5,AAA14/6,AAA15

*************************
*                       *
* STORAGE FOR L(1)      *
* CACHES                *
*                       *
*************************

       STORAGE    S$RID11,10/S$SID11,2/S$TID11,10/S$AID11,10
       STORAGE    S$RID12,10/S$SID12,2/S$TID12,10/S$AID12,10
       STORAGE    S$RID13,10/S$SID13,2/S$TID13,10/S$AID13,10
       STORAGE    S$RID14,10/S$SID14,2/S$TID14,10/S$AID14,10
       STORAGE    S$RID15,10/S$SID15,2/S$TID15,10/S$AID15,10

*************************
*                       *
* STORAGE FOR DEVICES   *
*                       *
*************************

       STORAGE    S$RID21,10/S$SID21,10/S$TID21,10
       STORAGE    S$RID22,10/S$SID22,10/S$TID22,10
       STORAGE    S$RID31,10/S$SID31,10/S$TID31,10
       STORAGE    S$RID32,10/S$SID32,10/S$TID32,10
       STORAGE    S$RID41,10/S$SID41,10/S$TID41,10
       STORAGE    S$RID42,10/S$SID42,10/S$TID42,10
*************************
*                       *
* STORAGE FOR REQ PROC  *
*                       *
*************************

       STORAGE    S$RIR2,10/S$SIR2,10/S$TIR2,10/S$AIR2,10/S$OIR2,10
       STORAGE    S$RIR3,10/S$SIR3,10/S$TIR3,10/S$AIR3,10/S$OIR3,10
       STORAGE    S$RIR4,10/S$SIR4,10/S$TIR4,10/S$AIR4,10/S$OIR4,10

*************************
*                       *
*  STORAGE FOR K1       *
*                       *
*************************

       STORAGE    S$ROK1,10/S$SOK1,10/S$TIK1,10/S$AIK1,10/S$OOK1,10

*************************
*                       *
```

```
*  STORAGE FOR K2,K3,K4  *
*                        *
**************************

       STORAGE     S$RIK2,10/S$SIK2,10/S$TIK2,10/S$AIK2,10/S$OIK2,10
       STORAGE     S$RIK3,10/S$SIK3,10/S$TIK3,10/S$AIK3,10/S$OIK3,10
       STORAGE     S$RIK4,10/S$SIK4,10/S$TIK4,10/S$AIK4,10/S$OIK4,10
       STORAGE     S$ROK2,10/S$SOK2,10/S$TOK2,10/S$AOK2,10/S$OOK2,10
       STORAGE     S$ROK3,10/S$SOK3,10/S$TOK3,10/S$AOK3,10/S$OOK3,10
       STORAGE     S$ROK4,10/S$SOK4,10/S$TOK4,10/S$AOK4,10/S$OOK4,10

**************************
*                        *
*  BOOLEAN VARIABLES     *
*                        *
**************************

**************************
*                        *
*  BV FOR READ-THROUGH   *
*                        *
**************************

RTOK2  BVARIABLE  FNU$GBUS*SNF$TIK1
RTOK3  BVARIABLE  FNU$GBUS*SNF$TIK1*SNF$TIK2
RTOK4  BVARIABLE  FNU$GBUS*SNF$TIK1*SNF$TIK2*SNF$TIK3

**************************
*                        *
*  BV FOR L(1)           *
*                        *
**************************

DKP1   BVARIABLE  FNU$LBUS1*SNF$ROK1
DKS1   BVARIABLE  FNU$LBUS1*SNF$SOK1
DKO1   BVARIABLE  FNU$LBUS1*SNF$OOK1
KDT11  BVARIABLE  FNU$LBUS1*SNF$TID11
KDT12  BVARIABLE  FNU$LBUS1*SNF$TID12
KDT13  BVARIABLE  FNU$LBUS1*SNF$TID13
KDT14  BVARIABLE  FNU$LBUS1*SNF$TID14
KDT15  BVARIABLE  FNU$LBUS1*SNF$TID15
KDA11  BVARIABLE  FNU$LBUS1*SNF$AID11
KDA12  BVARIABLE  FNU$LBUS1*SNF$AID12
KDA13  BVARIABLE  FNU$LBUS1*SNF$AID13
KDA14  BVARIABLE  FNU$LBUS1*SNF$AID14
KDA15  BVARIABLE  FNU$LBUS1*SNF$AID15

**************************
*                        *
*  BV FOR INTER LEVEL COM*
*                        *
**************************

KRR12  BVARIABLE  FNU$GBUS*SNF$RIK2
KKS12  BVARIABLE  FNU$GBUS*SNF$SIK2
```

```
KKO12  BVARIABLE  FNU$GBUS*SNF$SOK2
KKT21  BVARIABLE  FNU$GBUS*SNF$TIK1
KKA21  BVARIABLE  FNU$GBUS*SNF$AIK1
KKR23  BVARIABLE  FNU$GBUS*SNF$RIK3
KKS23  BVARIABLE  FNU$GBUS*SNF$SIK3
KKO23  BVARIABLE  FNU$GBUS*SNF$OIK3
KKT32  BVARIABLE  FNU$GBUS*SNF$TIK2
KKA32  BVARIABLE  FNU$GBUS*SNF$AIK2
KKR34  BVARIABLE  FNU$GBUS*SNF$RIK4
KKS34  BVARIABLE  FNU$GBUS*SNF$SIK4
KKO34  BVARIABLE  FNU$GBUS*SNF$OIK4
KKT43  BVARIABLE  FNU$GBUS*SNF$TIK3
KKA43  BVARIABLE  FNU$GBUS*SNF$AIK3

************************
*                      *
* BV FOR L(2) OPS      *
*                      *
************************

KPR2   BVARIABLE  FNU$LBUS2*SNF$RIR2
KRS2   BVARIABLE  FNU$LBUS2*SNF$SIR2
KRT2   BVARIABLE  FNU$LBUS2*SNF$TIR2
KRA2   BVARIABLE  FNU$LBUS2*SNF$AIR2
KPO2   BVARIABLE  FNU$LBUS2*SNF$OIR2
RDR21  BVARIABLE  FNU$LBUS2*SNF$RID21
RDS21  BVARIABLE  FNU$LBUS2*SNF$SID21
RDT21  BVARIABLE  FNU$LBUS2*SNF$TID21
RDR22  BVARIABLE  FNU$LBUS2*SNF$RID22
RDS22  BVARIABLE  FNU$LBUS2*SNF$SID22
RDT22  BVARIABLE  FNU$LBUS2*SNF$TID22
DKS2   BVARIABLE  FNU$LBUS2*SNF$SOK2
DKT2   BVARIABLE  FNU$LBUS2*SNF$TOK2
DKA2   BVARIABLE  FNU$LBUS2*SNF$AOK2
RKR2   BVARIABLE  FNU$LBUS2*SNF$ROK2
RKO2   BVARIABLE  FNU$LBUS2*SNF$OOK2
RKA2   BVARIABLE  FNU$LBUS2*SNF$AOK2

************************
*                      *
* BV FOR L(3) OPS      *
*                      *
************************

KRR3   BVARIABLE  FNU$LBUS3*SNF$RIR3
KRS3   BVARIABLE  FNU$LBUS3*SNF$SIR3
KRT3   BVARIABLE  FNU$LBUS3*SNF$TIR3
KRA3   BVARIABLE  FNU$LBUS3*SNF$AIR3
KPO3   BVARIABLE  FNU$LBUS3*SNF$OIR3
RDR31  BVARIABLE  FNU$LBUS3*SNF$RID31
RDS31  BVARIABLE  FNU$LBUS3*SNF$SID31
RDT31  BVARIABLE  FNU$LBUS3*SNF$TID31
RDR32  BVARIABLE  FNU$LBUS3*SNF$RID32
RDS32  BVARIABLE  FNU$LBUS3*SNF$SID32
RDT32  BVARIABLE  FNU$LBUS3*SNF$TID32
```

```
 DKS3   BVARIABLE   FNU$LBUS3*SNF$SOK3
 DKT3   BVARIABLE   FNU$LBUS3*SNF$TOK3
 DKA3   BVARIABLE   FNU$LBUS3*SNF$AOK3
 DKR3   BVARIABLE   FNU$LBUS3*SNF$ROK3
 DKA3   BVARIABLE   FNU$LBUS3*SNF$AOK3
 RKO3   BVARIABLE   FNU$LBUS3*SNF$OOK3

************************
*                      *
* BV FOR L(4) OPS      *
*                      *
************************

 KRR4   BVARIABLE   FNU$LBUS4*SNF$RIR4
 KRS4   BVARIABLE   FNU$LBUS4*SNF$SIR4
 KRO4   BVARIABLE   FNU$LBUS4*SNF$OIR4
 RDR41  BVARIABLE   FNU$LBUS4*SNF$RID41
 RDS41  BVARIABLE   FNU$LBUS4*SNF$SID41
 RDR42  BVARIABLE   FNU$LBUS4*SNF$RID42
 RDS42  BVARIABLE   FNU$LBUS4*SNF$SID42
 DKT4   BVARIABLE   FNU$LBUS4*SNF$TOK4
 DKA4   BVARIABLE   FNU$LBUS4*SNF$AOK4

************************
*                      *
*    MACROS            *
*                      *
************************

************************
*                      *
* MACRO -USE           *
* #A  FACILITY         *
* #B  USAGE TIME       *
*                      *
************************

 USE    STARTMACRO
        SEIZE       #A
        ADVANCE     #B
        ASSIGN      4+,#B
        RELEASE     #A
        ENDMACRO

************************
*                      *
*  MACRO - SEND        *
*                      *
* #A  FROM             *
* #B  TO               *
* #C  VIA              *
* #D  TRANSIT TIME     *
* #E  BV FOR SEND OP   *
*                      *
************************
```

```
 SEND   STARTMACRO
        TEST E      #E,1
        ENTER       #B
        SEIZE       #C
        ADVANCE     #D
        ASSIGN      4+,#D
        RELEASE     #C
        LEAVE       #A
        ENDMACRO

*************************
*                       *
*  MACRO - FINI         *
*                       *
*************************

 FINI   STARTMACRO
        MARK        3
        SAVEVALUE   NTXN+,1
        SAVEVALUE   SUMX+,V$TXNX
        SAVEVALUE   SUMW+,V$TXNW
        SAVEVALUE   SUMT+,V$TXNT
        SAVEVALUE   MRESP,V$MRESP
        ASSIGN      1,0
        ASSIGN      2,0
        ASSIGN      3,0
        ASSIGN      4,0
        ENDMACRO

**-----------------------------------------------------------------------*
*                                                                       *
*           BEGIN SIMULATION                                            *
*                                                                       *
**-----------------------------------------------------------------------*

        SIMULATE
*************************
*                       *
*   CPU #1              *
*                       *
*************************

        RMULT       3,5,7,9,11,13,15,17

 CPU1   GENERATE    ,,,X$MAXMP,,,F
 STAR1  PRIORITY    9                    SET HIGH P FOR NEW TXN
        MARK        2                    ARRIVAL TIME
        ASSIGN      1,1                  CPU ID
        TRANSFER    .X$NPEAD,WWW1,RRR1
 RRR1   TRANSFER    .X$PIN1,NIN11,EIN11

*************************
*                       *
* DATA IS IN DATA CACHE *
*                       *
```

```
*                         *
***************************

PIN11 ENTER     RID11           PUT TXN IN READ REQ BUFFER
USE   MACRO     DRP11,X$RDEX1   SEARCH AND READ CACHE
      LEAVE     RID11           FREE BUFFER
FINI  MACRO
      TRANSFER  ,STAR1          A NEW TXN

***************************
*                         *
* DATA IS NOT IN CACHE    *
*                         *
***************************

NIN11 ENTER     RID11           PUT IN READ REQ BUFFER
USE   MACRO     DRP11,X$REX     SEARCH DIRECTORY
      PRIORITY  0               RESET PRIORITY
SEND  MACRO     RID11,ROK1,LBUS1,X$DEXM,BV$DKR1

      TRANSFER  ,COMR           TO COMMON CODE FOR READ

***************************
*                         *
* WRITE REQUEST TO CACHE  *
*                         *
***************************

WWW1  ENTER     SID11           PUT TXN IN WRITE REQ BUFFER

USE   MACRO     DRP11,X$RDEX1    WRITE DATA IN CACHE

      PRIORITY  0               RESET TXN PRIORITY
SEND  MACRO     SID11,SOK1,LBUS1,X$DEX1,BV$DKS1

      SPLIT     1,COMW

FINI  MACRO

      TRANSFER  ,STAR1          A NEW TXN

***************************
*                         *
*   CPU #2                *
*                         *
***************************

CPU2  GENERATE  ,,,X$MAXMP,,,F
STAR2 PRIORITY  9               SET HIGH P FOR NEW TXN
      MARK      2               ARRIVAL TIME
      ASSIGN    1,2             CPU ID
      TRANSFER  .X$NREAD,WWW2,RRR2
RRR2  TRANSFER  .X$PIN1,NIN12,NIN12

***************************
```

```
*                          *
* DATA IS IN DATA CACHE    *
*                          *
****************************

RIN12 ENTER      RID12              PUT TXN IN READ REQ BUFFER
USE   MACRO      DRP12,X$RDEX1      SEARCH AND READ CACHE
      LEAVE      RID12              FREE BUFFER
FINI  MACRO
      TRANSFER   ,STAR2             A NEW TXN

****************************
*                          *
* DATA IS NOT IN CACHE     *
*                          *
****************************

NIN12 ENTER      RID12              PUT IN READ REQ BUFFER
USE   MACRO      DRP12,X$REX        SEARCH DIRECTORY
      PRIORITY   0                  RESET PRIORITY
SEND  MACRO      RID12,SOK1,LBUS1,X$BEXM,BV$DKB1

      TRANSFER   ,COMR              TO COMMON CODE FOR READ

****************************
*                          *
* WRITE REQUEST TO CACHE   *
*                          *
****************************

WWW2  ENTER      SID12              PUT TXN IN WRITE REQ BUFFER

USE   MACRO      DRP12,X$RDEX1      WRITE DATA IN CACHE

      PRIORITY   0                  RESET TXN PRIORITY
SEND  MACRO      SID12,SOK1,LBUS1,X$DEX1,BV$DKS1

      SPLIT      1,COMW

FINI  MACRO

      TRANSFER   ,STAR2             A NEW TXN

****************************
*                          *
*   CPU #3                 *
*                          *
****************************

CPU3  GENERATE   ,,,X$MAXMP,,,F
STAR3 PRIORITY   9                  SET HIGH P FOR NEW TXN
      MARK       2                  ARRIVAL TIME
      ASSIGN     1,3                CPU ID
      TRANSFER   .X$NREAD,WWW3,RRR3
RRR3  TRANSFER   .X$PIN1,NIN13,RIN13
```

```
*************************
*                       *
* DATA IS IN DATA CACHE *
*                       *
*************************

RIN13 ENTER     RID13             PUT TXN IN READ REQ BUFFER
USE   MACRO     DRP13,X$RDEX1     SEARCH AND READ CACHE
      LEAVE     RID13             FREE BUFFER
FINI  MACRO
      TRANSFER  ,STAR3            A NEW TXN

*************************
*                       *
* DATA IS NOT IN CACHE  *
*                       *
*************************

NIN13 ENTER     RID13             PUT IN READ REQ BUFFER
USE   MACRO     DRP13,X$REX       SEARCH DIRECTORY
      PRIORITY  0                 RESET PRIORITY
SEND  MACRO     RID13,ROK1,LBUS1,X$BEXM,BV$DKR1

      TRANSFER  ,COMR             TO COMMON CODE FOR READ

*************************
*                       *
* WRITE REQUEST TO CACHE*
*                       *
*************************

WWW3  ENTER     SID13             PUT TXN IN WRITE REQ BUFFER

USE   MACRO     DRP13,X$RDEX1      WRITE DATA IN CACHE

      PRIORITY  0                 RESET TXN PRIORITY

SEND  MACRO     SID13,SOK1,LBUS1,X$BEX1,BV$DKS1

      SPLIT     1,COMW

FINI  MACRO

      TRANSFER  ,STAR3

*************************
*                       *
*   CPU #4              *
*                       *
*************************

CPU4  GENERATE  ,,,X$MAXMP,,,F
STAR4 PRIORITY  9                 SET HIGH P FOR NEW TXN
      MARK      2                 ARRIVAL TIME
```

```
        ASSIGN      1,4                 CPU ID
        TRANSFER    .X$NREAD,WWW4,RRR4
 RRR4   TRANSFER    .X$PIN1,MIN14,RIN14

*************************
*                       *
* DATA IS IN DATA CACHE *
*                       *
*************************

 RIN14  ENTER       RID14               PUT TXN IN READ REQ BUFFER
 USE    MACRO       DRP14,X$RDEX1       SEARCH AND READ CACHE
        LEAVE       RID14               FREE BUFFER
 FINI   MACRO
        TRANSFER    ,STAR4              A NEW TXN

*************************
*                       *
* DATA IS NOT IN CACHE  *
*                       *
*************************

 MIN14  ENTER       RID14               PUT IN READ REQ BUFFER
 USE    MACRO       DRP14,X$REX         SEARCH DIRECTORY
        PRIORITY    0                   RESET PRIORITY
 SEND   MACRO       RID14,SOK1,LBUS1,X$BEXM,BV$DKR1

        TRANSFER    ,COMR               TO COMMON CODE FOR READ

*************************
*                       *
* WRITE REQUEST TO CACHE*
*                       *
*************************

 WWW4   ENTER       SID14               PUT TXN IN WRITE REQ BUFFER

 USE    MACRO       DRP14,X$RDEX1        WRITE DATA IN CACHE

        PRIORITY    0                   RESET TXN PRIORITY

 SEND   MACRO       SID14,SOK1,LBUS1,X$BEX1,BV$DKS1

        SPLIT       1,COMW

 FINI   MACRO

        TRANSFER    ,STAR4              A NEW TXN

*************************
*                       *
*   CPU #5              *
*                       *
*************************
```

```
CPU5  GENERATE   ,,,X$MAXMP,,,P
STAR5 PRIORITY   9                SET HIGH P FOR NEW TXN
      MARK       2                ARRIVAL TIME
      ASSIGN     1,5              CPU ID
      TRANSFER   .X$NREAD,WWW5,RRR5
RRR5  TRANSFER   .X$PIN1,NIN15,RIN15

*************************
*                       *
* DATA IS IN DATA CACHE *
*                       *
*************************

RIN15 ENTER      RID15            PUT TXN IN READ REQ BUFFER
USE   MACRO      DRP15,X$RDEX1    SEARCH AND READ CACHE
      LEAVE      RID15            FREE BUFFER
FINI  MACRO
      TRANSFER   ,STAR5           A NEW TXN

*************************
*                       *
* DATA IS NOT IN CACHE  *
*                       *
*************************

NIN15 ENTER      RID15            PUT IN READ REQ BUFFER
USE   MACRO      DRP15,X$REX      SEARCH DIRECTORY
      PRIORITY   0                RESET PRIORITY
SEND  MACRO      RID15,ROK1,LBUS1,X$BEXM,DV$DKR1

      TRANSFER   ,COMR            TO COMMON CODE FOR READ

*************************
*                       *
* WRITE REQUEST TO CACHE*
*                       *
*************************

WWW5  ENTER      SID15            PUT TXN IN WRITE REQ BUFFER

USE   MACRO      DRP15,X$RDEX1     WRITE DATA IN CACHE

      PRIORITY   0                RESET TXN PRIORITY

SEND  MACRO      SID15,SOK1,LBUS1,X$BEX1,DV$DKS1

      SPLIT      1,COMW

FINI  MACRO

      TRANSFER   ,STAR5           A NEW TXN

*-------------------------------------------------------------*
*                                                             *
*   COMMON CODE FOR READ REQUEST                              *
```

```
*                                                                     *
*---------------------------------------------------------------------*

 COMR   ASSIGN      11,0

 USE    MACRO       KRP1,X$KEX

 SEND   MACRO       ROK1,RIK2,GBUS,X$BEXM,BV$KKR12

 USE    MACRO       KRP2,X$KEX

 SEND   MACRO       RIK2,RIR2,LBUS2,X$BEXM,BV$KRR2

 USE    MACRO       RRP2,X$REX

        TRANSFER    .X$PIN2,NIN2,RIN2
 NIN2   ASSIGN      11,0

 SEND   MACRO       RIR2,ROK2,LBUS2,X$BEXM,BV$RKR2

 USE    MACRO       KRP2,X$KEX

 SEND   MACRO       ROK2,RIK3,GBUS,X$BEXM,BV$KKR23

 USE    MACRO       KRP3,X$KEX

 SEND   MACRO       RIK3,RIR3,LBUS3,X$BEXM,BV$KRR3

 USE    MACRO       RRP3,X$REX

        TRANSFER    .X$PIN3,NIN3,RIN3
 NIN3   ASSIGN      11,0

 SEND   MACRO       RIR3,ROK3,LBUS3,X$BEXM,BV$RKR3

 USE    MACRO       KRP3,X$KEX

 SEND   MACRO       ROK3,RIK4,GBUS,X$BEXM,BV$KKR34

 USE    MACRO       KRP4,X$KEX

 SEND   MACRO       RIK4,RIR4,LBUS4,X$BEXM,BV$KRR4

 USE    MACRO       RRP4,X$REX

        TRANSFER    ,RIN4

*---------------------------------------------------------------------*
*                                                                     *
*  READ DATA IS FOUND IN L(2)                                         *
*                                                                     *
*---------------------------------------------------------------------*

 RIN2   TRANSFER    .5,RRR21,RRR22
```

```
*************************
*                       *
* DATA IS IN D21        *
*                       *
*************************

 RRR21 ASSIGN    11,0
 SEND  MACRO     RIR2,RID21,LBUS2,X$BEXM,BV$PDR21
 USE   MACRO     DRP21,X$DEX2
 SEND  MACRO     RID21,TOK2,LBUS2,X$BEX1,BV$DKT2
       TRANSFER  ,RTF2

*************************
*                       *
* DATA IS IN D22        *
*                       *
*************************

 RRR22 ASSIGN    11,0
 SEND  MACRO     RIR2,RID22,LBUS2,X$DEXM,BV$RDR22
 USE   MACRO     DRP22,X$DEX2
 SEND  MACRO     RID22,TOK2,LBUS2,X$BEX1,BV$DKT2
       TRANSFER  ,RTF2

*************************
*                       *
* READ-THROUGH TO L(1)  *
*                       *
*************************

 RTF2  ASSIGN    11,0
 USE   MACRO     KRP2,X$KEX
 SEND  MACRO     TOK2,TIK1,GBUS,X$BEX1,BV$RTOK2

*-----------------------------------------------------------------*
*                                                                 *
* STORE DATA INTO L(1) AS RESULT OF A READ-THROUGH                *
*                                                                 *
*-----------------------------------------------------------------*

 STOR1 ASSIGN    11,0
 USE   MACRO     KRP1,X$KEX
       SPLIT     1,PN$WICHW,1
```

```
         TERMINATE

*************************
*                       *
* RT STORE INTO D11     *
*                       *
*************************

 WWW11 ASSIGN     11,0

 SEND  MACRO      TIK1,TID11,LBUS1,X$BEX1,BV$KDT11

 USE   MACRO      DRP11,X$DEX1

       TRANSFER   .X$POV1,NOV11,OVL11
 NOV11 LEAVE      TID11

 FINI  MACRO

       TRANSFER   ,STAR1

 OVL11 SPLIT      1,OVF11

 FINI  MACRO

       TRANSFER   ,STAR1
 OVF11 ASSIGN     11,0

 SEND  MACRO      TID11,OOK1,LBUS1,X$BEXM,BV$DKO1

       TRANSFER   ,OVL1

*************************
*                       *
* RT STORE INTO D12     *
*                       *
*************************

 WWW12 ASSIGN     11,0

 SEND  MACRO      TIK1,TID12,LBUS1,X$BEX1,BV$KDT12

 USE   MACRO      DRP12,X$DEX1

       TRANSFER   .X$POV1,NOV12,OVL12
 NOV12 LEAVE      TID12

 FINI  MACRO

       TRANSFER   ,STAR2

 OVL12 SPLIT      1,OVF12

 FINI  MACRO
```

```
         TRANSFER   ,STAR2
OVF12    ASSIGN     11,0

SEND     MACRO      TID12,OOK1,LBUS1,X$BEXM,BV$DKO1

         TRANSFER   ,OVL1

*************************
*                       *
* RT STORE INTO D13     *
*                       *
*************************

WWW13    ASSIGN     11,0

SEND     MACRO      TIK1,TID13,LBUS1,X$BEX1,BV$KDT13

USE      MACRO      DRP13,X$DEX1

         TRANSFER   .X$POV1,NOV13,OVL13
NOV13    LEAVE      TID13

FINI     MACRO

         TRANSFER   ,STAR3

OVL13    SPLIT      1,OVF13

FINI     MACRO

         TRANSFER   ,STAR3
OVF13    ASSIGN     11,0

SEND     MACRO      TID13,OOK1,LBUS1,X$BEXM,BV$DKO1

         TRANSFER   ,OVL1

*************************
*                       *
* RT STORE INTO D14     *
*                       *
*************************

WWW14    ASSIGN     11,0

SEND     MACRO      TIK1,TID14,LBUS1,X$BEX1,BV$KDT14

USE      MACRO      DRP14,X$DEX1

         TRANSFER   .X$POV1,NOV14,OVL14
NOV14    LEAVE      TID14

FINI     MACRO

         TRANSFER   ,STAR4
```

```
 OVL14 SPLIT      1,OVF14

 FINI  MACRO

       TRANSFER   ,STAR4
 OVF14 ASSIGN     11,0

 SEND  MACRO      TID14,OOK1,LBUS1,X$BEXM,BV$DKO1

       TRANSFER   ,OVL1

*************************
*                       *
* RT STORE INTO D15     *
*                       *
*************************

 WWW15 ASSIGN     11,0

 SEND  MACRO      TIK1,TID15,LBUS1,X$BEX1,BV$KDT15

 USE   MACRO      DRP15,X$DEX1

       TRANSFER   .X$POV1,NOV15,OVL15
 NOV15 LEAVE      TID15

 FINI  MACRO

       TRANSFER   ,STAR5

 OVL15 SPLIT      1,OVF15

 FINI  MACRO

       TRANSFER   ,STAR5
 OVF15 ASSIGN     11,0

 SEND  MACRO      TID15,OOK1,LBUS1,X$BEXM,BV$DKO1

       TRANSFER   ,OVL1

*************************
*                       *
* HANDLE OVF FROM L(1)  *
*                       *
*************************

 OVL1  ASSIGN     11,0

 USE   MACRO      KRP1,X$KEX

 SEND  MACRO      OOK1,OIK2,GBUS,X$BEXM,BV$KKO12

 USE   MACRO      KRP2,X$KEX
```

```
 SEND  MACRO     OIK2,OIR2,LBUS2,X$BEXM,BV$KRO2
 USE   MACRO     RRP2,X$REX
       LEAVE     OIB2
       TERMINATE
*-------------------------------------------------------------------*
*                                                                   *
*  READ DATA IS FOUND IN L(3)                                       *
*                                                                   *
*-------------------------------------------------------------------*
 RIN3  TRANSFER  .5,RRR31,RRR32
*************************
*                       *
* DATA IS IN D31        *
*                       *
*************************
 RRR31 ASSIGN    11,0
 SEND  MACRO     RIR3,RID31,LBUS3,X$BEXM,BV$RDR31
 USE   MACRO     DRP31,X$DEX3
 SEND  MACRO     RID31,TOK3,LBUS3,X$BEX2,BV$DKT3
       TRANSFER  ,RTF3
*************************
*                       *
* DATA IS IN D32        *
*                       *
*************************
 RRR32 ASSIGN    11,0
 SEND  MACRO     RIR3,RID32,LBUS3,X$BEXM,BV$RDR32
 USE   MACRO     DRP32,X$DEX3
 SEND  MACRO     RID32,TOK3,LBUS3,X$BEX2,BV$DKT3
       TRANSFER  ,RTF3
*************************
*                       *
* RT TO L(1) AND L(2)   *
*                       *
*************************
 RTF3  ASSIGN    11,0
```

```
USE   MACRO     KRP3,X$KEX

      TEST E    BV$PTOK3,1
      ENTER     TIK1
      ENTER     TIK2
      SEIZE     GBUS
      ADVANCE   X$BEX2
      ASSIGN    4+,X$BEX2
      RELEASE   GBUS
      LEAVE     TOK3
      SPLIT     1,STOR1
      SPLIT     1,STOR2
      TERMINATE

*-------------------------------------------------------------*
*                                                             *
* STORE DATA INTO L(2) AS RESULT OF A READ-THROUGH            *
*                                                             *
*-------------------------------------------------------------*

STOR2 ASSIGN    11,0

USE   MACRO     KRP2,X$KEX

SEND  MACRO     TIK2,TIR2,LBUS2,X$BEX2,BV$KRT2

USE   MACRO     RRP2,X$REX

      SPLIT     1,OVH2
      TRANSFER  .5,SSS21,SSS22

**************************
*                        *
* STORE INTO D21         *
*                        *
**************************

SSS21 ASSIGN    11,0

SEND  MACRO     TIR2,TID21,LBUS2,X$BEX2,BV$RDT21

USE   MACRO     DRP21,X$DEX2

      LEAVE     TID21
      TERMINATE

**************************
*                        *
* STORE INTO D22         *
*                        *
**************************

SSS22 ASSIGN    11,0
```

```
 SEND  MACRO      TIR2,TID22,LBUS2,X$BEX2,BV$RDT22

 USE   MACRO      DRP22,X$DEX2

       LEAVE      TID22
       TERMINATE

*************************
*                       *
* OVERFLOW HANDLING     *
*                       *
*************************

 OVR2  TRANSFER   .X$BOV2,NOVL2,OVL2
 OVL2  TEST E     BV$RKO2,1
       ENTER      OOK2
       SEIZE      LBUS2
       ADVANCE    X$BEXM
       ASSIGN     4+,X$BEXM
       RELEASE    LBUS2

 SEND  MACRO      OOK2,OIK3,GBUS,X$BEXM,BV$KKO23

 USE   MACRO      KRP3,X$KEX

 SEND  MACRO      OIK3,OIR3,LBUS3,X$BEXM,BV$KRO3

 USE   MACRO      RRP3,X$REX

       LEAVE      OIR3
 NOVL2 TERMINATE

*-----------------------------------------------------------------*
*                                                                 *
* READ DATA IS FOUND IN L(4)                                      *
*                                                                 *
*-----------------------------------------------------------------*

 RIN4  TRANSFER   .5,RRR41,RRR42

*************************
*                       *
* DATA IS IN D41        *
*                       *
*************************

 RRR41 ASSIGN     11,0

 SEND  MACRO      RIR4,RID41,LBUS4,X$BEXM,BV$RDR41

 USE   MACRO      DRP41,X$DEX4

 SEND  MACRO      RID41,TOK4,LBUS4,X$BEX3,BV$DKT4

       TRANSFER   ,RTF4
```

```
*************************
*                       *
* DATA IS IN D42        *
*                       *
*************************

BRR42 ASSIGN    11,0

SEND  MACRO     RIR4,RID42,LBUS4,X$BEXM,BV$RDR42

USE   MACRO     DR242,X$DEX4

SEND  MACRO     RID42,TOK4,LBUS4,X$BEX3,BV$DKT4

      TRANSFER  ,RTF4

*************************
*                       *
* RT TO L(1),L(2),L(3)  *
*                       *
*************************

RTF4  ASSIGN    11,0

USE   MACRO     KRP4,X$KEX

      TEST E    BV$RTOK4,1
      ENTER     TIK1
      ENTER     TIK2
      ENTER     TIK3
      SEIZE     GBUS
      ADVANCE   X$BEX3
      ASSIGN    4+,X$BEX3
      RELEASE   GBUS
      LEAVE     TOK4
      SPLIT     1,STOR1
      SPLIT     1,STOR2
      SPLIT     1,STOR3
      TERMINATE

*-----------------------------------------------------------------*
*                                                                 *
*  STORE INTO L(3) AS A RESULT OF READ-THROUGH                    *
*                                                                 *
*-----------------------------------------------------------------*

STOR3 ASSIGN    11,0

USE   MACRO     KRP3,X$KEX

SEND  MACRO     TIK3,TIR3,LBUS3,X$BEX3,BV$KRT3

USE   MACRO     RRP3,X$REX
```

```
         SPLIT     1,OVH3
         TRANSFER  .5,SSS31,SSS32

**************************
*                        *
* STORE INTO D31         *
*                        *
**************************

SSS31    ASSIGN    11,0

 SEND    MACRO     TIR3,TID31,LBUS3,X$BEX3,BV$RDT31

 USE     MACRO     DRP31,X$DEX3

         LEAVE     TID31
         TERMINATE

**************************
*                        *
* STORE INTO D32         *
*                        *
**************************

 SSS32 ASSIGN      11,0

 SEND    MACRO     TIR3,TID32,LBUS3,X$BEX3,BV$RDT32

 USE     MACRO     DRP32,X$DEX3

         LEAVE     TID32
         TERMINATE

**************************
*                        *
* OVERFLOW HANDLING      *
*                        *
**************************

 OVH3    TRANSFER  .X$PCV3,NOVL3,OVL3
 OVL3    TEST E    BV$KKO3,1
         ENTER     OCK3
         SEIZE     LBUS3
         ADVANCE   X$BEXM
         ASSIGN    4+,X$BEXM
         RELEASE   LBUS3

 SEND    MACRO     OOK3,OIK4,GBUS,X$BEXM,BV$KKO34

 USE     MACRO     KRP4,X$KEX

 SEND    MACRO     OIK4,OIR4,LBUS4,X$BEXM,BV$KRO4

 USE     MACRO     RRP4,X$REX
         LEAVE     OIR4
```

```
 NOVL3 TERMINATE
*-------------------------------------------------------------------*
*                                                                   *
*  COMMON CODE FOR STORE-BEHIND                                     *
*                                                                   *
*-------------------------------------------------------------------*

 COMW  ASSIGN     11,0
 USE   MACRO      KRP1,X$KEX
 SEND  MACRO      SOK1,SIK2,GBUS,X$BEX1,BV$KKS12
 USE   MACRO      KRP2,X$KEX
 SEND  MACRO      SIK2,SIR2,LBUS2,X$BEX1,BV$KRS2
 USE   MACRO      RRP2,X$REX
       TRANSFER   .5,SWS21,SWS22

**************************
*                        *
* SB WRITE INTO D21      *
*                        *
**************************

 SWS21 ASSIGN     11,0
 SEND  MACRO      SIR2,SID21,LBUS2,X$BEX1,BV$RDS21
 USE   MACRO      DRP21,X$DEX2
 SEND  MACRO      SID21,SOK2,LBUS2,X$BEX2,BV$DKS2
       SPLIT      1,STB23
       ENTER      AOK2
       TRANSFER   ,ACK21

**************************
*                        *
* SB WRITE INTO D22      *
*                        *
**************************

 SWS22 ASSIGN     11,0
 SEND  MACRO      SIR2,SID22,LBUS2,X$BEX1,BV$RDS22
 USE   MACRO      DRP22,X$DEX2
 SEND  MACRO      SID22,SOK2,LBUS2,X$BEX2,BV$DKS2
       SPLIT      1,STB23
```

```
        ENTER       AOK2
        TRANSFER    ,ACK21
*-------------------------------------------------------------------*
*                                                                   *
* STORE-BEHIND TO L(3)                                              *
*                                                                   *
*-------------------------------------------------------------------*
 STB23  ASSIGN      11,0
 USE    MACRO       KRP2,X$KEX
 SEND   MACRO       SOK2,SIK3,GBUS,X$BEX2,BV$KKS23
 USE    MACRO       KRP3,X$KEX
 SEND   MACRO       SIK3,SIR3,LBUS3,X$BEX2,BV$KRS3
 USE    MACRO       RRP3,X$REX
        TRANSFER    .5,SWS31,SWS32
**************************
*                        *
* SB WRITE INTO D31      *
*                        *
**************************
 SWS31  ASSIGN      11,0
 SEND   MACRO       SIR3,SID31,LBUS3,X$BEX2,BV$RDS31
 USE    MACRO       DRP31,X$BEX3
 SEND   MACRO       SID31,SOK3,LBUS3,X$DEX3,BV$DKS3
        SPLIT       1,STB34
        ENTER       AOK3
        TRANSFER    ,ACK32
**************************
*                        *
* SB WRITE INTO D32      *
*                        *
**************************
 SWS32  ASSIGN      11,0
 SEND   MACRO       SIR3,SID32,LBUS3,X$BEX2,BV$RDS32
 USE    MACRO       DRP32,X$DEX3
 SEND   MACRO       SID32,SOK3,LBUS3,X$DEX3,BV$DKS3
```

```
        SPLIT     1,STB34
        ENTER     AOK3
        TRANSFER  ,ACK32

*-------------------------------------------------------------*
*                                                             *
*  STORE-BEHIND TO L(4)                                       *
*                                                             *
*-------------------------------------------------------------*

 STB34 ASSIGN     11,0
 USE   MACRO      KRP3,X$KEX
 SEND  MACRO      SOK3,SIK4,GBUS,X$BEX3,BV$KKS34
 USE   MACRO      KRP4,X$KEX
 SEND  MACRO      SIK4,SIR4,LBUS4,X$BEX3,BV$KRS4
 USE   MACRO      RRP4,X$REX
       TRANSFER   .5,SWS41,SWS42

************************
*                      *
* SB WRITE INTO D41    *
*                      *
************************

 SWS41 ASSIGN     11,0
 SEND  MACRO      SIR4,SID41,LBUS4,X$BEX3,BV$RDS41
 USE   MACRO      DRP41,X$DEX4
 SEND  MACRO      SID41,AOK4,LBUS4,X$BEXM,BV$DKA4
       TRANSFER   ,ACK43

************************
*                      *
* SB WRITE INTO D42    *
*                      *
************************

 SWS42 ASSIGN     11,0
 SEND  MACRO      SIR4,SID42,LBUS4,X$BEX3,BV$RDS42
 USE   MACRO      DRP42,X$DEX4
 SEND  MACRO      SID42,AOK4,LBUS4,X$BEXM,BV$DKA4
       TRANSFER   ,ACK43
```

```
*-------------------------------------------------------------------*
*                                                                   *
*  ACK FROM L(4) TO L(3)                                            *
*                                                                   *
*-------------------------------------------------------------------*

 ACK43 ASSIGN     11,0
 USE   MACRO      KRP4,X$KEX
 SEND  MACRO      AOK4,AIK3,GBUS,X$BEXM,BV$KKA43
 USE   MACRO      KRP3,X$KEX
 SEND  MACRO      AIK3,AIR3,LBUS3,X$BEXM,BV$KRA3
 ..
 USE   MACRO      RRP3,X$REX

*************************
*                       *
* FORWARD THE ACK UP    *
*                       *
*************************

 SEND  MACRO      AIR3,AOK3,LBUS3,X$BEXM,BV$RKA3
 USE   MACRO      KRP3,X$KEX
 SEND  MACRO      AOK3,AIK2,GBUS,X$BEXM,BV$KKA32
 USE   MACRO      KRP2,X$KEX
 SEND  MACRO      AIK2,AIR2,LBUS2,X$BEXM,BV$KRA2
 USE   MACRO      RRP2,X$REX
       LEAVE      AIR2
       TERMINATE

*-------------------------------------------------------------------*
*                                                                   *
* ACK FROM L(3) TO L(2)                                             *
*                                                                   *
*-------------------------------------------------------------------*

 ACK32 ASSIGN     11,0
 USE   MACRO      KRP3,X$KEX
 SEND  MACRO      AOK3,AIK2,GBUS,X$BEXM,BV$KKA32
 USE   MACRO      KRP2,X$KEX
 SEND  MACRO      AIK2,AIR2,LBUS2,X$BEXM,BV$KRA2
```

```
 USE    MACRO       DRP2,X$REX
 SEND   MACRO       AIR2,AOK2,LBUS2,X$BEXM,BV$RKA2
        TRANSFER    ,ACK21
*-------------------------------------------------------------------*
*                                                                   *
*  ACK FROM L(2) TO L(1)                                            *
*                                                                   *
*-------------------------------------------------------------------*
 ACK21  ASSIGN      11,0
 USE    MACRO       KRP2,X$KEX
 SEND   MACRO       AOK2,AIK1,GBUS,X$BEXM,BV$KKA21
 USE    MACRO       KRP1,X$KEX
        SPLIT       1,FN$WICHA,1
        TERMINATE
**************************
*                        *
* ACK HANDLED BY D11     *
*                        *
**************************
 AAA11  ASSIGN      11,0
 SEND   MACRO       AIK1,AID11,LBUS1,X$BEXM,BV$KDA11
 USE    MACRO       DRP11,X$REX
        LEAVE       AID11
        TERMINATE
**************************
*                        *
* ACK HANDLED BY D12     *
*                        *
**************************
 AAA12  ASSIGN      11,0
 SEND   MACRO       AIK1,AID12,LBUS1,X$BEXM,BV$KDA12
 USE    MACRO       DRP12,X$REX
        LEAVE       AID12
        TERMINATE
**************************
```

```
*                          *
* ACK HANDLED BY D13       *
*                          *
****************************

 AAA13 ASSIGN     11,0

 SEND  MACRO      AIK1,AID13,LBUS1,X$BEXM,BV$KDA13

 USE   MACRO      DRP13,X$REX

       LEAVE      AID13
       TERMINATE

****************************
*                          *
* ACK HANDLED BY D14       *
*                          *
****************************

 AAA14 ASSIGN     11,0

 SEND  MACRO      AIK1,AID14,LBUS1,X$BEXM,BV$KDA14

 USE   MACRO      DRP14,X$REX

       LEAVE      AID14
       TERMINATE

****************************
*                          *
* ACK HANDLED BY D15       *
*                          *
****************************

 AAA15 ASSIGN     11,0

 SEND  MACRO      AIK1,AID15,LBUS1,X$BEXM,BV$KDA15

 USE   MACRO      DRP15,X$REX

       LEAVE      AID15
       TERMINATE

*--------------------------------------------------------------------*
*                                                                    *
*  SIMULATION CONTROL                                                *
*                                                                    *
*--------------------------------------------------------------------*

       GENERATE   X$TIMER
       TERMINATE  1
       START      1
       END
```